Architecture Principles – A Regulative Perspective on Enterprise Architecture.

Conference Paper · January 2007

Source: DBLP

CITATIONS
51

READS
705

4 authors, including:

Stijn Hoppenbrouwers
Hogeschool Arnhem and Nijmegen
154 PUBLICATIONS **2,344** CITATIONS

SEE PROFILE

Henderik Alex Proper
Luxembourg Institute of Science and Technology (LIST)
432 PUBLICATIONS **5,979** CITATIONS

SEE PROFILE

Architecture principles –
A regulative perspective on enterprise architecture

P. (Patrick) van Bommel[1], P.G. (Pieter) Buitenhuis[2],
S.J.B.A. (Stijn) Hoppenbrouwers[1] and H.A. (Erik) Proper[1]

[1]Institute for Computing and Information Sciences, Radboud University Nijmegen
Toernooiveld 1, 6525 ED Nijmegen, The Netherlands
{P.vanBommel,S.Hoppenbrouwers,E.Proper}@cs.ru.nl

[2]Ordina
Ringwade 1, 3439 LM Nieuwegein, The Netherlands
Pieter.Buitenhuis@ordina.nl

Abstract: Increasingly, organizations make use of enterprise architectures to direct the development of the enterprise as a whole and its IT portfolio in particular. In this paper we investigate the regulative nature of enterprise architecture. We aim to develop a fundamental understanding of the regulative needs that underly an enterprise architecture, and then take these needs as a starting point to arrive at requirements on the language (architecture principles) used to denote enterprise architectures. We furthermore discuss the process of formulating principles as well as their semantics.

1 Introduction

Increasingly, organizations make use of enterprise architectures to direct the development of the enterprise as a whole and its IT portfolio in particular [Lo05]. These developments are fuelled by requirements such as the Clinger-Cohan Act in the USA[1], which force government bodies to provide an IT architecture.

The term architecture has been used in the field of IT since the 1960's. In the early days it was used to refer to the principles underlying the design of computer hardware and operating systems. This led to the use of the term computer architecture. Later, when software applications became larger and larger, Mary Shaw and David Garlan coined the term software architecture [SG96]. This notion of architecture deals with the key design principles underlying software artefacts. In the 1980's and 1990's people became aware that the development of IT (information technology) should be done in tandem with the development of the context in which it was to be used. This led to the identification of the so-called Business/IT alignment problem [PB89, TC93, HV93]. Solving the Business/IT alignment problem requires organisations to align human, organisational, informational and technological aspects of systems. Quite early on, architecture was also introduced

[1]http://www.cio.gov/Documents/it_management_reform_act_Feb_1996.html

as a means to further alignment, and thus analyse and solve Business/IT alignment problems [Zac87, TC93, Boa99b]. The Business/IT alignment problem requires the alignment of information technology, consisting of software and hardware, to the other 'technologies' used in a business. This has led to the use of the term architecture at the enterprise level [BL96, Boa99a, Lo05]. Enterprise architectures typically bring together a business, information, application and technology perspective on (parts of) an enterprise.

Ultimately, enterprise architectures are a means to an end. The Software Engineering Institute from Carnegy Mellon University identifies the following potential uses [BCK98] for architectural desriptions:

- It is a vehicle for communication among stakeholders.
- It captures early design decisions, both functional aspects as well as quality aspects.
- It describes the global structure decided upon in the architecture, also structures further development.
- It is a transferable abstraction of a system.

Many different perspectives exist on what architecture, in an IT context, actually is. Even though some consensus exists, the field of enterprise architecture is still in its infancy. However, the widespread use of enterprise architecture illustrates that organizations feel a profound need to steer their development (including their IT portfolio) and that they look towards enterprise architecture as a means to fulfill this need. When studying the many existing definitions on architecture [BL96, SG96, BCK98, Boa99a, IEE00, TOG04, Lo05, xAF06], one can discern two important perspectives on architecture:

Regulative perspective – Architecture is regarded as a prescriptive notion limiting the design freedom with regards to the design of a system. When taking this perspective one will focus on principles, leading to rules/principles limiting designers in their design freedom.

Designing perspective – Architectures are actual specifications of high level system designs focussing on 'architecturally relevant' design decisions. When taking this perspective, one typically produces architectural models that describe the design of actual system artefacts.

These two perspectives are complementary in that the regulative perspective accommodates for the need to steer and direct developments, whereas the second perspective supports the need to gain insight into an enterprise's design while also providing guidance to designers of enterprise systems [Lo05].

In this paper, we focus on the regulative perspective. In taking a regulative perspective on enterprise architecture, we are primarily concerned with its ability to steer the over-all enterprise/system development within a large organization (enterprise). A more specific way of expressing this is to state that "*Architecture serves the purpose of constraining design space*" [xAF06]. In most (enterprise) architecture approaches, this constraining/regulating

is done by means of so-called architecture principles [IEE00, TOG04]. The aforementioned Clinger-Cohen act also requires the architecture to be specified in terms of a set of principles. Such architecture principles usually take the form of informal statements such as (taken from [TOG04]):

> *Users have access to the data necessary to perform their duties; therefore, data is shared across enterprise functions and organizations.*

According to the TOGAF architecture framework [TOG04], "*Principles are general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way in which an organization sets about fulfilling its mission.*" Such principles typically address concerns of the key stakeholders within an organization. In the example case, a stakeholder may be highly concerned about the organization's ability to flexibly deploy their workforce over different work locations.

While several sources attribute a pivotal role to principles, a precise definition of the concept of principles as well as the mechanisms and procedures needed to turn them into an effective regulatory means still lacks. Both IEEE [IEE00] and TOGAF [TOG04] position principles as a means to guide the design and evolution of systems, while xAF [xAF06] essentially equates (enterprise) architecture to a set of principles. In any case, no clear definition of principles and associated mechanisms and procedures are given.

When considering the definitions reported in literature [TC93, IEE00, TOG04, xAF06], and the definitions used by several practitioners, three key perspectives on principles can be discerned:

Inherent laws – These are essentially properties of (classes of) a system that can be observed and validated.

Conceptual parallels are the laws of nature, law of requisite variety, laws of social behavior, etc.

Imposed laws – Like inherent laws, they are properties of (classes of) a system that can be validated. However, imposed laws also require mechanisms to enforce them.

Imposed laws typically address concerns of stakeholders. Some of these concerns may be raised by emergent laws having a negative impact on the system being designed.

Examples are: societal laws, policies and regulations within organizations, etc.

Guidelines – Desired properties that are so concrete that they offer guidelines to make operational behavior fit imposed laws.

For example: "use your car's cruise control" is an advisable property to abide by that provides guidance in obeying the law concerning maximum speeds on roads.

In this paper we mainly focus on the last two perspectives on principles.

The remainder of this paper is structured as follows. In section 2 we aim to build up a better understanding of the regulative role of enterprise architectures. This is followed

in section 3 by a discussion of requirements on the language of architecture principles, as a means to operatonalize the regulative ability of enterprise architectures. Section 4 then continues by discussing the semantics of principles, in other words, their regulative impact. Before concluding, section 5 discusses an approach to the formulation of architecture principles based on practical experiences and some experiments.

2 Enterprise architecture as a regulative mechanism

As mentioned in the introduction, this paper takes a regulative perspective on enterprise architecture. This role comes primarily to the fore in the role of architecture principles [IEE00, TOG04, xAF06]. In [xAF06], enterprise architecture is equated to a set of architecture principles, which are to "limit design freedom", thus regulating the freedom of an enterprise's designers.

The aim of this section is to briefly investigate the regulative needs that underly an enterprise architecture. When indeed taking the perspective that an enterprise architecture is a regulative means, one must also agree (at least from a rational perspective) that there is some regulative need motivating the use of the means.

Enterprises have stakeholders. For example: owners, sponsors, people working in the enterprise, clients, etc. Let S be a stakeholder of an enterprise E, then it is fair to assume that S has some goals $\mathsf{Goals}(S)$ which are potentially impacted on by the behaviour of E. From the perspective of these goals, the enterprise E has an ideal behaviour. This behaviour can refer to all aspects of an enterprise, be it the operational processes, financial aspects, labour issues, adaptability, etc.

The actual attainment of E's ideal behaviour from the perspective of $\mathsf{Goals}(S)$ may be influenced by both internal and external factors [BMM06]. These potential 'impacts' may spark stakeholder S into (trying to) regulate enterprise E and/or its influencers. Needless to say that S will not be the only stakeholder, and the desires of S to regulate E may indeed conflict with the regulatory desires of other stakeholders.

Given some influencer F, a risk assessment (see also [BMM06]) may show that F has a potential undesired impact on $\mathsf{Goals}(S)$, in other words there is a set of risks $\mathsf{Risks}(F, S)$ influencer F poses to the goals of stakeholder S by potentially influencing E. Each of these risks can be quantified in terms of its possible impact $\mathsf{Impact}(R)$ and probability of occurrence $\mathsf{Probability}(R)$, with expected impact: $\mathsf{Impact}(R) \times \mathsf{Probability}(R)$. Note: the impact *might* be approximated using an amount of money, but ultimately relates to the impact of a stakeholder's goals.

If the expexted impact is high enough, stakeholder S will be motivated to introduce regulations to prevent R from occurring. If M is some (set of) regulation(s) aimed at R, then its benefit can be assessed by determining:

$$\mathsf{Benefit}(R, M) \triangleq (\mathsf{Impact}(R) \times \mathsf{Probability}(R)) - (\mathsf{Impact}(R|M) \times \mathsf{Probability}(R|M))$$

where $\mathsf{Impact}(R|M)$ and $\mathsf{Probability}(R|M)$ represents the possible impact and probability of occuring of risk R with regulation M in place. If $\mathsf{Costs}(M)$ are the costs of

deploying regulation M, then the actual effectiveness of M can be thought of as being the ratio:

$$\mathsf{Benefit}(R, M)/\mathsf{Costs}(M)$$

The costs can, again, be expressed in terms of *money*, but should yet again be regarded as a negative impact on the (satisfaction of) goals a stakeholder is striving for.

Admittedly, this cost/benefit framework is as yet far from practical. This is in line with the observation that the regulative role of enterprise architecture is also still in its infancy. At the same time, however, the desire to use enterprise architecture for such purposes is paramount.

In making this kind of risk assessment more explicit, the field of enterprise architecting may borrow from the field of security [RF07], where security risks are assessed and the effectivness of security regulations are evaluated against these risks. The aim of this paper is not to develop such a risk assessment, although we subscribe to its necessity, but rather to explore requirements on a principles language. In further research, however, we will indeed invest in a more elaborate cost/benefit analysis approach providing rationalization of principles. In doing so, we will start by evaluating principles (and their enforcement mechanisms) in the context of the scenario's underlying the risks identified by the stakeholders, with the aim of assessing the contribution of these principles in terms of reduced impact and/or reduced chances of the risks occurring.

3 Requirements on principles

In this section we focus on the goals of, and requirements on, a modelling language for architecture principles. When taking the position that architecture principles embody the regulative role of enterprise architecture, then we can this as a starting point to indentify requirements on a language to express architecture principles. In order to arrive at such requirements, it is important to first make explicit what the goals of the language are [HPW05b]. Without these goals it is difficult to pinpoint the requirements for the language in total. On their turn, these requirements are a mandatory input to formulation of modelling concepts in the language and their requirements [HPW05b, PVH05].

In a survey, in terms of a number of in-depth interviews conducted among a number of experienced enterprise architects [Bui07], the following goals where expressed:

Regulative architecture – The language should be designed in such a way that it enables an architect formulate a prescriptive/regulative architecture.

Supportive; not restrictive – The language should not act as a straitjacket but should rather should architects in formulating the regulative architecture. It is important that the modelling language does not hinder the (creative!) architecting process itself.

Architect independent – Since architecture principles need to be communicated among architects, to stakeholders, and system designers, the language as such should not

be specific to one architect. Even though this may sound obvious, it is still common practice among enterprise architects to come up with one's own language. There is, as yet, not much consensus about such a language.

Approach independent – A good modelling language can be used in different development approaches. For example, like UML, ER and ORM can be used in system development methods such as SDM, DSDM, RAD and XP. This should also be the case for this an architecture principles language; it should be independent of the architecting approach used.

A means of communicating and steering – Architecture is positioned as a communication and a steering device. This must be taken in consideration when designing this modelling language. The steering and communication goals may lead to conflicts. For example, a nice marketing line (such as Nokia's '*Connecting People*') may be suitable to communicate a basic idea, while not being specific enough to give enough steering.

Based on the above goals, and also as part of the survey, the following requirements were deduced [Bui07]:

Facilitating role – The architecting process and not the modelling language should have the most impact on the architecture itself. The architect should not be (too) restricted by the language in formulating the architecture. This is why the modelling language should give the architect some tools and means (in the form of formulation guidelines) to formulate an architecture better. It is then the choice of the architect to use those guidelines. Because an architecture is subject to evolution, it is necessary for the language to be able to support the different stages (from sketch to definitive formulation) of the architecting process.

Syntactically complete – The modelling language needs to contain all the different concepts that are used by architects when formulating a regulative architecture. This implies that it must be clear which concepts are used by architects and how they relate to each other. Because each architecture should always be considered in its context, those concepts should be used in the language as well. It is clear that an incomplete modelling language can not produce a complete architecture. The completeness of concepts in the language should be resolved on syntactical level.

Contains reference architecture Some architects have pleaded for a modelling language involving numerous examples. This has two advantages:

- it gives the architect possible means to formulate a better architecture and
- it will be possible to create a reference architecture.

Reference architectures play a role of importance in giving the field of enterprise architecting a more mature status. It is still common practice for architects to start from scratch for each new project. This implies that architectures are very often not proven in practice which does not give the client the guarantee that the architecture will work in practise.

Contains also (semi-)formalised language – Regulative architectures are currently primarily based on natural language. Interviewed architects do see the advantages of a (semi-)formalised architecture, but claim as well that the architecture should also be communicated to the 'normal' stakeholders. The modelling language should therefore not only facilitate (semi-)formalised language, but also the 'normal' natural language. The architect should not be forced to write formalised statements.

The formalised concepts can be used by the architect to make a check on completeness and (formalised) linguistic aspects. Because formalised statements are less ambiguous, it's very advisable to use them with project members who can handle those statements.

When using natural language, there is also a distinction to be made in the degree of abstractness in the formulation. A statement for higher management will normally be different (more concise, more simple, more catchy) then for a project member.

The language is then capable of making different visualizations of the same architecture, focused on the type of stakeholder. Formulating an architecture on a formalised and non formalised level is also consistent with the two mail goals of architecture. A (semi-)formalised language supports primarily the steering function, the natural language the communication aspect.

Terminological framework; improving the communication – To improve the communication and decrease the ambiguity of the architecture, it's very important to have a shared term framework between all stakeholders. Such a framework should be based and focused on the stakeholders. However, it is now impossible to insert a fixed term framework in this modelling language because of the architect independence goal. This is also consistent with the requirement that the modelling language should not be to prescriptive.

In addition to these requiremens voiced by practitioners, additional requirements can be found in literature. In their Architecture Framework (TOGAF), the Open Group [TOG04] lists five criteria (which are also based on practical experiences) that distinguish a good set of principles:

Understandable – The underlying tenets can be quickly grasped and understood by individuals throughout the organization. The intention of the principle is clear and unambiguous, so that violations, whether intentional or not, are minimized.

Robust – Enable good quality decisions about architectures and plans to be made, and enforceable policies and standards to be created. Each principle should be sufficiently definitive and precise to support consistent decision making in complex, potentially controversial, situations.

Complete – Every potentially important principle governing the management of information and technology for the organization is defined. The principles cover every situation perceived.

Consistent – Strict adherence to one principle may require a loose interpretation of another principle. The set of principles must be expressed in a way that allows a balance of interpretations. Principles should not be contradictory to the point where adhering to one principle would violate the spirit of another. Every word in a principle statement should be carefully chosen to allow consistent yet flexible interpretation.

Stable – Principles should be enduring, yet able to accommodate changes. An amendment process should be established for adding, removing, or altering principles after they are ratified initially.

The above requirements are requirements on a set of principles and are not requirements on the modelling language. However, these requirements on good principles do underline the need for:

1. a clear semantics of principles, enabling a.o. consistency checks of sets of principles,
2. have a syntax which is understandable by domain experts and not just architects and engineers.

The TOGAF requirements also imply requirements on the process of formulating and maintaining sets of architecture principles.

A further source of requirements on architecture principles and/or a language for modelling architecture principles can be found in the business rules manifesto [Ros03]. Business rules are also forms of regulations that should both have a precise meaning while also be understandable to domain experts/stakeholders. The business rules manifesto lists a set of requirements / principles that should be met by business rules and their application. Most of these requirements also apply to architecture principles. Some key (from an architecture principle perspective) requirements from the business rules manifesto are:

3.2 Terms express business concepts; facts make assertions about these concepts; rules constrain and support these facts.

3.3 Rules must be explicit. No rule is ever assumed about any concept or fact.

4.1 Rules should be expressed declaratively in natural-language sentences for the business audience.

5.1 Business rules should be expressed in such a way that they can be validated for correctness by business people.

5.2 Business rules should be expressed in such a way that they can be verified against each other for consistency.

5.3 Formal logics, such as predicate logic, are fundamental to well-formed expression of rules in business terms, as well as to the technologies that implement business rules.

7.1 Rules define the boundary between acceptable and unacceptable business activity.

8.4 "More rules" is not better. Usually fewer "good rules" is better.

Most of these requirements are in line with the requirements put forward by TOGAF.

4 Semantics of principles

The TOGAF (as well as business rules manifesto) requirement of rules being *consistent* and at the same time being *understandable* by domain experts and stakeholders, provides an interesting challenge in the construction of a principles modelling language.

In [BHPW06, CJN+07], we have studied the use of a semi-formal language to represent principles. The language used stems from the field of information modelling, where languages such as ConQuer, Lisa-D and RIDL [Mee82, HPW93, Pro94, BH96, HPW05a] have been used to formulate constraints, rules and queries and reason about these.

Consider as an example the following TOGAF principle:

Common use applications – "Development of applications used across the enterprise is preferred over the development of duplicate applications which are only provided to a particular organization."

A domain analysis and formalization leads to:
If an Application A [**that** is used in **an** Organization O] results from **some** Development,
and this Application A **is not** a duplicate of **another** Application
[**that** is used in **another** Organization **than** O], **then that** Development
is preferred by **the** Enterprise **that** includes **both** Organizations and **both** Applications.
The **boldface** words are the keywords of the language, while the non-boldface are domain specific words.

An important question in this example is the way one would have to measure when one application is a duplicate of another application. In making such principles SMART[2], proper mechanisms should be defined to determine whether one application is a duplicate of another one, or more appropriately, whether one set of applications is a duplicate of another set. And more generally, in the aspect system Business one would like to measure when one process is a duplicate of another process, in order to detect process and organizational redundancy.

5 Formulating principles

In [NBP07] we have reported on research efforts into the collaborative formulation of policies/regulations such as architecture principles. This work mainly focussed on the collaborative aspects of such processes. Note that the TOGAF requirements of robustness, completeness and stability of architecture principles have not yet been taken into account in this work. The experience report given in [OP07] did take such requirements into consideration. In [BHPW06, CJN+07] the possible effect of using a semi-formal formal modelling language in the formulation of principles was studied.

[2] Specific, Measurable, Achievable, Relevant, Time-bound; a common mnemonic used in project management.

Based on these experiments and experiences, we suggest adhering to the following process-structure in formulating architecture principles:

Assess needs – An assessments of the regulative needs, which can be used as motivations for the principles to be formulated and their enforcement.

1. (*Collaborative*) In a collaborative session involving stakeholders, sponsors, domain experts and architects, potential regulative needs (issues/concerns/risks) should be gathered. In addition, goals should be formulated upon these regulative needs are to be assessed, and criteria should be formulated regarding the desired longevity of any measures (architecture principles) formulated that are to address these needs.
2. (*Expert-driven*) Risk assessment experts should then assess/judge the identified regulative needs. This assessment should take the form of a risk analysis as suggested in section 2.
3. (*Collaborative*) The stakeholders and the sponsors (of the regulative measures to be taken) should then decide which of these regulative needs they want to see addressed.

Formulate principles – The definition of a set of principles that are to be deployed.

1. (*Collaborative*) Based on the (seleced) regulative needs, a mix of stakeholders, domain experts and architects should, collaboratively, formulate a set of architecture principles which would address these needs.
2. (*Expert-driven*) The resulting candidate principles should then be pinned down more precisely by a small number of domain experts together with the architects. This group should also assess the longevity of the principles in terms of the criteria produced during the needs assessment, as well as determine more specific consequences of the principle, and clearly identify/quantify the possible contribution of the principle towards the regulative needs.
3. (*Collaborative*) The list of refined principles should then be put to the vote. The domain experts, stakeholders and sponsors should select which principles are to be deployed.

Prepare deployment – For each principle a deployment scenario should be formulated.

1. (*Collaborative*) Given the list of selected architecture principles, more refined criteria for the assessment of possible strategies to deploy/enforce these principles should be formulated.
2. (*Expert-driven*) Domain experts and architects should define a number of possible strategies for the deployment/enforcement of the select architecture principles. Each of these strategies should also be evaluated in terms of their costs/benefits.
3. (*Collaborative*) From the list of available scenario's for the deployment of principles, the stakeholders, domain experts and sponsors should select the ones they see as most effective and beneficial.

The above procedure iterates between a collabarative and expert-driven mode. Some tasks should be done collaboratively so as to warrant committment from, and understanding by, all relevant parties involved. Some other tasks require a focussed effort by a limited number of experts.

Note that the above sketched process structure by no means intends to imply a specific length/duration/size of a specific formulation process. Depending on the requirements of specific situation, such a process may/should require only a single day or even weeks to complete.

6 Conclusions and further research

In this paper we have investigated the regulative nature of enterprise architecture. The work reported is part of an ongoing effort to develop a fundamental understanding of the regulative needs that underly an enterprise architecture, and use this understanding in the development of a language for architecture principles as well as a situational procedure/strategy for the formulation of such principles.

In our remaining research activities regarding architecture principles, we identify the following key challenges:

Language – How are principles to be expressed, i.e. in what type of language? Are there any hard syntactic or semantic restrictions that are to be imposed, or would this be too restrictive? What could be reasons to (not) impose particular such restrictions? How can understandability be optimally safeguarded at a linguistic level, and how much investment in this is justified in view of quality demands on principle formulations? (How) can SMARTness be increased based on language restrictions?

Regulative needs – How can the regulative needs underlying principles be better quantified? How can one predict the possitive contributions of enforcing a principle towards these needs?

Formulation strategies – Given requirements on the product of the principles creation process, what does such a process look like? Are situational adjustments of the process required or is it possible to define a truly generic process? Apart from the question what semantic and syntactic requirements can be posed for principles, there are pragmatic matters to be addressed. Principles are required to be understood, agreed upon, and committed to by appropriate (groups of) stakeholders. These should be viewed as results of the process, alongside the actual principle formulations [BHP07].

Deployment strategies – Enforcement and guidance during the design and development of new systems (and new versions), but also strategies for dealing with legacy systems. How to translate principles and their measuring mechanisms to guidelines? Can the impact of principles be estimated reliably beforehand?

References

[BCK98] L. Bass, P.C. Clements, and R. Kazman. *Software Architecture in Practice*. Addison Wesley, Reading, Massachusetts, USA, 1998.

[BH96] A.C. Bloesch and T.A. Halpin. ConQuer: A Conceptual Query Language. In B. Thalheim, editor, *Proceedings of the 15th International Conference on Conceptual Modeling (ER'96), Cottbus, Germany, EU*, volume 1157 of *Lecture Notes in Computer Science*, pages 121–133, Berlin, Germany, EU, October 1996. Springer.

[BHP07] P. van Bommel, S.J.B.A. Hoppenbrouwers, and H.A. (Erik) Proper. QoMo: A Modelling Process Quality Framework based on SEQUAL. In H.A. (Erik) Proper, T.A. Halpin, and J. Krogstie, editors, *Proceedings of the Workshop on Exploring Modeling Methods for Systems Analysis and Design (EMMSAD'07), held in conjunctiun with the 19th Conference on Advanced Information Systems (CAiSE'07), Trondheim, Norway*, pages 118–127. Tapir Academic Press, Trondheim, Norway, 2007.

[BHPW06] P. van Bommel, S.J.B.A. Hoppenbrouwers, H.A. (Erik) 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 – OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, GADA, MIOS+INTEROP, ORM, PhDS, SeBGIS, SWWS, and WOSE 2006*, Lecture Notes in Computer Science, Montpellier, France, EU, October/November 2006. Springer, Berlin, Germany, EU.

[BL96] M. Blechar and M. Light. Enterprise Information Architectures. Technical Report R–450–131, GartnerGroup – ADM, February 1996.

[BMM06] BMM Team. Business Motivation Model (BMM) Specification. Technical Report dtc/06–08–03, Object Management Group, Needham, Massachusetts, USA, August 2006.

[Boa99a] B.H. Boar. *Constructing Blueprints for Enterprise IT architectures*. Wiley, New York, New York, USA, 1999.

[Boa99b] B.H. Boar. *Practical steps for aligning information technology with business strategies*. Wiley, New York, New York, USA, 1999.

[Bui07] P.G. Buitenhuis. Fundamenten van het principle (Foundations of principles). Master's thesis, Institute for Computing and Information Sciences, Radboud University Nijmegen, Nijmegen, The Netherlands, EU, March 2007. In Dutch.

[CJN+07] G.J.N.M. Chorus, Y.H.C. Janse, C.J.P. Nellen, S.J.B.A. Hoppenbrouwers, and H.A. (Erik) Proper. Formalizing Architecture Principles using Object–Role Modelling. Technical Report ICIS–R07006, Radboud University Nijmegen, February 2007.

[HPW93] A.H.M. ter Hofstede, H.A. (Erik) Proper, and Th.P. van der Weide. Formal definition of a conceptual language for the description and manipulation of information models. *Information Systems*, 18(7):489–523, October 1993.

[HPW05a] S.J.B.A. Hoppenbrouwers, H.A. (Erik) Proper, and Th.P. van der Weide. Fact Calculus: Using ORM and Lisa–D to Reason About Domains. In R. Meersman, Z. Tari, and P. Herrero, editors, *On the Move to Meaningful Internet Systems 2005: OTM Workshops – OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, GADA, MIOS+INTEROP, ORM, PhDS, SeBGIS, SWWS, and WOSE 2005*, volume 3762 of *Lecture Notes in Computer Science*, pages 720–729, Agia Napa, Cyprus, EU, October/November 2005. Springer, Berlin, Germany, EU.

[HPW05b] S.J.B.A. Hoppenbrouwers, H.A. (Erik) Proper, and Th.P. van der Weide. Understanding the Requirements on Modelling Techniques. In O. Pastor and J. Falcao e Cunha, editors, *17th International Conference on Advanced Information Systems Engineering, CAiSE 2005, Porto, Portugal, EU*, volume 3520 of *Lecture Notes in Computer Science*, pages 262–276, Berlin, Germany, EU, June 2005. Springer–Verlag.

[HV93] J.C. Henderson and N. Venkatraman. Strategic alignment: Leveraging information technology for transforming organizations. *IBM Systems Journal*, 32(1):4–16, 1993.

[IEE00] Recommended Practice for Architectural Description of Software Intensive Systems. Technical Report IEEE P1471–2000, The Architecture Working Group of the Software Engineering Committee, Standards Department, IEEE, Piscataway, New Jersey, USA, September 2000.

[Lo05] M.M. Lankhorst and others. *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Berlin, Germany, EU, 2005.

[Mee82] R. Meersman. The RIDL Conceptual Language. Technical report, International Centre for Information Analysis Services, Control Data Belgium, Inc., Brussels, Belgium, EU, 1982.

[NBP07] J. Nabukenya, P. van Bommel, and H.A. (Erik) Proper. Collaborative IT Policy-making as a means of achieving Business-IT Alignment. In B. Pernici and J.A. Gulla, editors, *Proceedings of the Workshop on Business/IT Alignment and Interoperability (BUSITAL'07), held in conjunctiun with the 19th Conference on Advanced Information Systems (CAiSE'07), Trondheim, Norway*, pages 461–468. Tapir Academic Press, Trondheim, Norway, 2007.

[OP07] M. Op 't Land and H.A. (Erik) 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.

[PB89] M.M. Parker and R.J. Benson. Enterprisewide Information Management: State–of–the–art Strategic Planning. *Journal of Information Systems Management*, (Summer):14–23, 1989.

[Pro94] H.A. (Erik) Proper. ConQuer–92 – The revised report on the conceptual query language LISA–D. Technical report, Asymetrix Research Laboratory, University of Queensland, Brisbane, Queensland, Australia, 1994.

[PVH05] H.A. (Erik) Proper, A.A. Verrijn–Stuart, and S.J.B.A. Hoppenbrouwers. Towards Utility–based Selection of Architecture–Modelling Concepts. In S. Hartmann and M. Stumptner, editors, *Proceedings of the Second Asia–Pacific Conference on Conceptual Modelling (APCCM2005), Newcastle, New South Wales, Australia*, volume 42 of *Conferences in Research and Practice in Information Technology Series*, pages 25–36, Sydney, New South Wales, Australia, January 2005. Australian Computer Society.

[RF07] S. Raval and A. Fichadia. *Risks, Controls and Security – Concepts and Applications*. Wiley, New York, New York, USA, 2007.

[Ros03] R.G. Ross, editor. *Business Rules Manifesto*. Business Rules Group, November 2003. Version 2.0.

[SG96] M. Shaw and D. Garlan. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice–Hall, Englewood Cliffs, New Jersey, USA, 1996.

[TC93] D. Tapscott and A. Caston. *Paradigm Shift – The New Promise of Information Technology*. McGraw–Hill, New York, New York, USA, 1993.

[TOG04] The Open Group. *TOGAF – The Open Group Architectural Framework*, 2004.

[xAF06] xAF working group. Extensible Architecture Framework version 1.1 (formal edition). Technical report, 2006.

[Zac87] J.A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3), 1987.