Inf Syst Front (2006) 8:103–113
DOI 10.1007/s10796-006-7975-x

The many faces of architectural descriptions

Danny Greefhorst · Henk Koning · Hans van Vliet

Abstract In recent years architecture has acquired recognition as playing a pivotal role in change processes. Despite this recognition, describing architecture has proven to be difficult. Architecture frameworks have been defined to address this problem. However, there are many of them, and together they leave us with seemingly contradicting terminology. What are the underlying forces that caused people to create so many different frameworks? What do these frameworks teach us about the essence of architecting? Where do I start to select or create a framework for my current project? With these questions in mind we set out to perform a comparison of existing architecture frameworks. We ended up with a deeper understanding of the function of a framework, and "discovered" nine fundamental dimensions that seem to underlie architectural thinking. These "base dimensions" can be used to clarify the meaning of individual architecture documents independent of the framework they originate from, and they can be helpful in defining new architecture frameworks or situational architecture descriptions. In this paper we also relate our findings to IEEE 1471, which is another important generalisation of existing frameworks.

Keywords Architecture frameworks · Architectural dimensions · IEEE 1471

D. Greefhorst (✉)
IBM Business Consulting Services, The Netherlands
e-mail: greefhorst@nl.ibm.com

H. Koning
Freelance Researcher, The Netherlands
e-mail: koning@arc-it.nl

H. van Vliet
Vrije Universiteit, Amsterdam, The Netherlands
e-mail: hans@cs.vu.nl

Introduction

Architecture in IT has gained acceptance as a means to guide IT change processes. Although people tend to disagree on the exact definition, architecture can be seen as the high-level structure of a system. It describes fundamental aspects of the system, and guides the persons that actually design and build the system. Architecture needs to be described in a document: an architectural description. Also, the architectural description needs to be structured into manageable "chunks" that each addresses a number of aspects of the architecture. There is no universal agreement on the "chunks" that an architectural description should consist of. Architecture frameworks should provide guidance in this area, but the problem is that there are so many of them. Even when there is agreement on the use of a framework, the peculiarities of a specific project often make it necessary to deviate from the framework. IT-architecture consultants who work in varying circumstances have to spend extra time to get acquainted with local templates, and it may take some time before all the meanings are clearly understood. The trade of architecting is visually manifested mainly by the frameworks. They are the signs of mastery achieved. Persons who are new to the field, like junior architects, and who see so many diverging frameworks, ask themselves "what is going on here?". Can you imagine a doctor in a hospital saying "I have 18 ways to record your case in my files".

These observations motivated us to investigate frameworks for architectural descriptions, and try to discover their fundamental structure. Architecture frameworks order architectural descriptions along one or more axes, and typically visualize the resulting architectural space spanned by these axes. A cell in this *n*-dimensional space denotes an architectural description that corresponds to the characteristics of the accompanying column and row. We call these axes "dimensions", and call specific columns or rows in these dimensions "values". Further analysis of these dimensions led to the identification of nine "base dimensions", that are the foundation for the dimensions found in architecture frameworks. We will describe these base dimensions, and illustrate their relationship to the dimensions as they occur in existing architecture frameworks. The latter are typically a combination of one or more "base dimensions".

The main contribution of this paper is the insight that architectural "dimensions" should be made explicit, and are based on a number of "base dimensions". It is not our intention to introduce a new framework. Although we studied quite a number of frameworks, we do not contend that our list of base dimensions is complete. Also, the values within the base dimensions presented are merely described to illustrate the base dimensions. Finally, we did not strive to make a complete survey of architecture frameworks, but only to have a solid enough basis for analyzing the "logic of architectural frameworks". Readers are urged to use the base dimensions presented as a reference point to position individual architecture documents or to better understand the essentials of existing architecture frameworks.

An important milestone in the field of architecture descriptions is ANSI/IEEE Std 1471 (IEEE Std 1471-2000, 2000), which was published in 2000. We will refer to this standard as IEEE 1471. IEEE 1471 proposes to structure architecture descriptions in views which are directly related to stakeholder concerns. In this paper we point to some strengths and weaknesses of this approach and we show that our findings are complementary to IEEE 1471.

This paper is organised in three sections. In the first section we describe the current situation of the architecture frameworks. We will give a short description of two architecture frameworks, just to introduce the notion of an architecture framework, and the concept of "dimension" to readers unfamiliar with them. We then list a number of architecture frameworks, and the dimensions we discovered in them. This is followed by an analysis, leading to some general observations and essentials of architecting. In the second section we elaborate on the concept of "dimension", and propose a list of base dimensions in architecture. We illustrate the usage of the base dimensions with an example. In the third section we relate our work to IEEE 1471. We conclude with a short recap and acknowledgements and references.

Architecture frameworks

Architecture frameworks offer a standard approach to architecture. This approach may encompass a model for architectural descriptions, as well as a method to produce them. Some architecture frameworks focus on the architectural descriptions, while others focus on the method. In this paper we are mainly interested in the way architecture frameworks approach architectural descriptions, and structure them into one or more dimensions. Further analysis of the space of architecture frameworks shows that they can be divided into two categories: enterprise-class frameworks and application-class frameworks.

Enterprise-class frameworks are aimed at business units, complete organisations or even industry sectors. These frameworks often have multiple dimensions, potentially leading to a large number of architectural models. An enterprise architectural information base may contain many separately maintained documents. Examples of enterprise-class frameworks are the Zachman Framework for Information Systems Architecture (ISA) (Zachman, 1987; Sowa and Zachman, 1992), the Information Framework (IFW) (Evernden, 1996), The Open Group Architecture Framework (TOGAF) (The Open Group, 2003), Integrated Architecture Framework (IAF) (Goedvolk and Rijsenbrij, 1999) and Methodology for Architecture Description (MAD) (Meinema, 1999).

Application-class frameworks describe the architecture of a specific (software) application or a group of similar applications, and typically comprise a small number of architectural models. The information in application-class frameworks is often more fine-grained than the information in enterprise-class frameworks. Well-known application-class frameworks are the 4+1 model (Kruchten, 1995), the framework of Siemens (Hofmeister et al., 1995, 2000), and the 2+2 model of the Vrije Universiteit (Lassing et al., 2001).

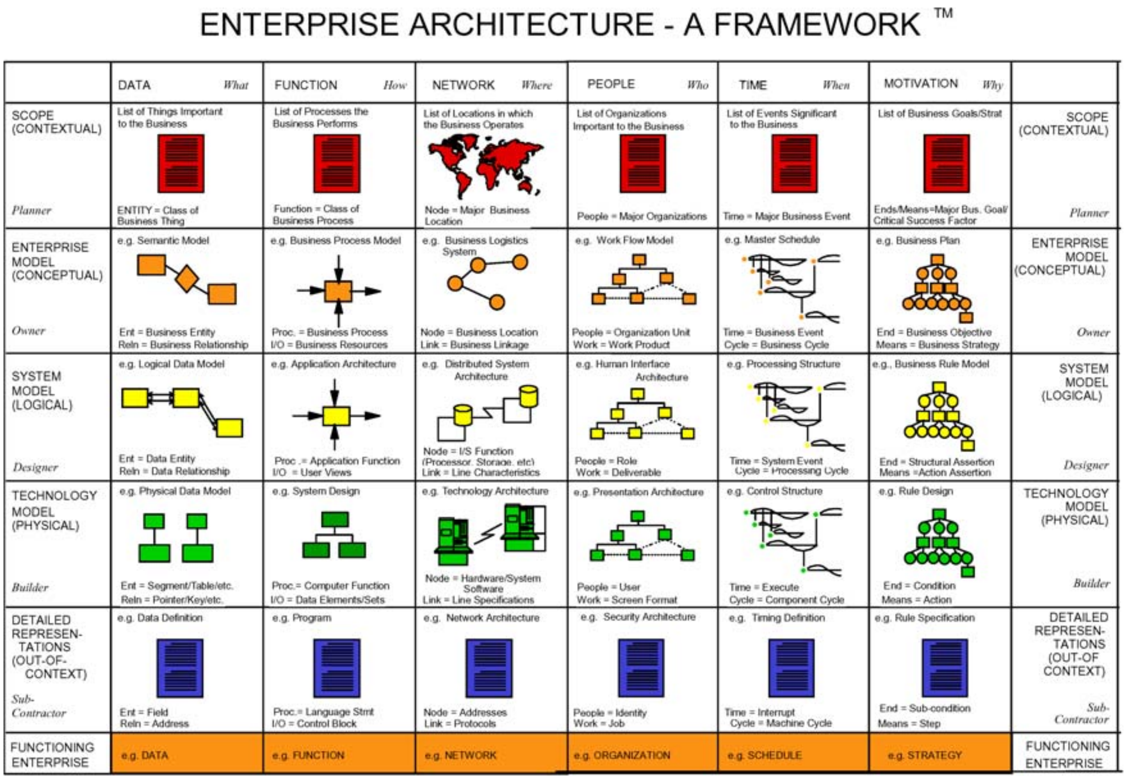
The following paragraph describes the Zachman framework and the 4+1 model in more detail as typical examples of enterprise-class and application-class frameworks, respectively. It also illustrates the concept of dimension.

Showcases

Zachman

The foundation for enterprise-class frameworks was laid by John Zachman in his 1987 article (Zachman, 1987) in which he describes a framework for the architecture of information systems. His idea was that architecture for information systems could be inspired by architecture in more mature engineering disciplines. He saw that the architectural models in these engineering disciplines showed a lot of similarities and could be combined in a generic model. Zachman recognised two dimensions: perspectives of specific target audiences and the types of architectural descriptions.

Potential perspectives are those of: the planner, the owner, the designer, the builder and the subcontractor of an information system. Later on, Zachman gave these perspectives more logical names, and they were labelled the contextual, conceptual, logical, physical and out-of-context perspectives. The



ENTERPRISE ARCHITECTURE - A FRAMEWORK ™

	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*	
SCOPE (CONTEXTUAL) *Planner*	List of Things Important to the Business ENTITY = Class of Business Thing	List of Processes the Business Performs Function = Class of Business Process	List of Locations in which the Business Operates Node = Major Business Location	List of Organizations Important to the Business People = Major Organizations	List of Events Significant to the Business Time = Major Business Event	List of Business Goals/Strat Ends/Means=Major Bus. Goal/ Critical Success Factor	SCOPE (CONTEXTUAL) *Planner*
ENTERPRISE MODEL (CONCEPTUAL) *Owner*	e.g. Semantic Model Ent = Business Entity Reln = Business Relationship	e.g. Business Process Model Proc. = Business Process I/O = Business Resources	e.g. Business Logistics System Node = Business Location Link = Business Linkage	e.g. Work Flow Model People = Organization Unit Work = Work Product	e.g. Master Schedule Time = Business Event Cycle = Business Cycle	e.g. Business Plan End = Business Objective Means = Business Strategy	ENTERPRISE MODEL (CONCEPTUAL) *Owner*
SYSTEM MODEL (LOGICAL) *Designer*	e.g. Logical Data Model Ent = Data Entity Reln = Data Relationship	e.g. Application Architecture Proc .= Application Function I/O = User Views	e.g. Distributed System Architecture Node = I/S Function (Processor, Storage, etc) Link = Line Characteristics	e.g. Human Interface Architecture People = Role Work = Deliverable	e.g. Processing Structure Time = System Event Cycle = Processing Cycle	e.g., Business Rule Model End = Structural Assertion Means =Action Assertion	SYSTEM MODEL (LOGICAL) *Designer*
TECHNOLOGY MODEL (PHYSICAL) *Builder*	e.g. Physical Data Model Ent = Segment/Table/etc. Reln = Pointer/Key/etc.	e.g. System Design Proc.= Computer Function I/O = Data Elements/Sets	e.g. Technology Architecture Node = Hardware/System Software Link = Line Specifications	e.g. Presentation Architecture People = User Work = Screen Format	e.g. Control Structure Time = Execute Cycle = Component Cycle	e.g. Rule Design End = Condition Means = Action	TECHNOLOGY MODEL (PHYSICAL) *Builder*
DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor*	e.g. Data Definition Ent = Field Reln = Address	e.g. Program Proc.= Language Stmt I/O = Control Block	e.g. Network Architecture Node = Addresses Link = Protocols	e.g. Security Architecture People = Identity Work = Job	e.g. Timing Definition Time = Interrupt Cycle = Machine Cycle	e.g. Rule Specification End = Sub-condition Means = Step	DETAILED REPRESEN-TATIONS (OUT-OF CONTEXT) *Sub-Contractor*
FUNCTIONING ENTERPRISE	e.g. DATA	e.g. FUNCTION	e.g. NETWORK	e.g. ORGANIZATION	e.g. SCHEDULE	e.g. STRATEGY	FUNCTIONING ENTERPRISE

Fig. 1 Zachman framework for enterprise architecture

out-of-context perspective denotes that at this level parts are typically fabricated outside the larger context in which they are used.

The types of description dimension finds its origin in Zachman's observation that the same elementary questions of what, how, where, who, when and why can always be answered in different contexts. For information systems these questions are translated to data, function, location, people, time and motivation. The other observation was that both dimensions could vary independently, leading to $5 * 6 = 30$ different kinds of architectural models for one information system. In the framework these models are depicted in a matrix with columns for the types of description and rows for the perspectives (see Fig. 1).

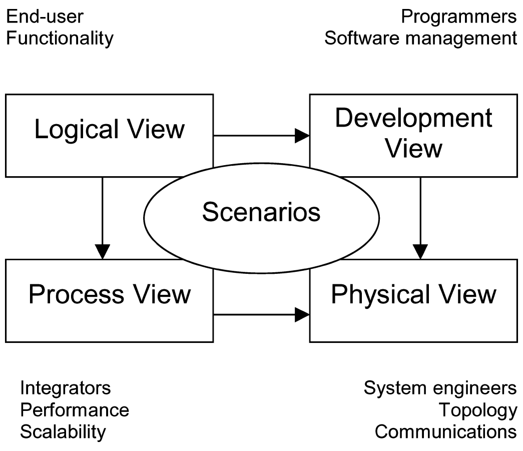


Fig. 2 The 4+1 model

4+1

A well-known application-class framework is the 4+1 model (Kruchten, 1995) for software (see Fig. 2).

In contrast with the enterprise-class frameworks, this framework only has one dimension, which is not named explicitly. Like the Zachman framework, the views relate to different stakeholders and their concerns. There are four views, namely the logical, development, process and physical view. These views have a recognisable relationship with users (classes), developers (packages and files), integrators (processes, messages) and system engineers (nodes and networks). The fifth view contains scenarios that describe how the elements in the other views co-operate.

Overview

We now offer a summarized overview of architecture frameworks, and other architecture classifications we found (see Table 1). For each framework we list the source, the dimensions and the values in the dimensions. The dimensions are depicted as the rows next to the framework. A division of

Table 1 Existing architecture frameworks

Framework	Source	Dimension	Values
2+2 model	(Lassing et al., 2001)		Context, Technical Infrastructure, Conceptual, Development
4+1 model	(Kruchten, 1995)		Logical, Process, Development, Physical, Scenarios
ADS	(Youngs et al., 1999)	Aspects	Functional, Operational
		Level	Specified, Physical
ARIS	(Scheer, 1992)		Organizational, Data, Control, Function, Product/Service
Boar	(Boar, 1998)		Infrastructure, Data, Applications, Organization
			Inventory, Principles, Models, Standards
CIMOSA	(ESPRIT, 1993)	Instantiation	Generic, Partial, Particular
		Views	Function, Information, Resource, Organisation
		Derivation	Requirements Definition, Design Specification, Implementation Description
DYA	(Wagter et al., 2001)		Business (Product, Process, Organisation), Information (Data, Application), Technical (Middleware, Platform, Network)
			Common Principles, Policies, Models
Evernden Eight	(Evernden, 2002)	Types of information	
		Levels of understanding	
		Types of representation	
		Levels of transition	
		Types of knowledge	
		Levels of responsibility	
		Types of process	
		Meta levels	
Gartner	(Rosser, 2002)	Scope	Multi enterprise Grid, Enterprise, Business Process, Brick
			Context, Concept, Logical
			Now, less than 2 years, 2 to 5 years
GEM	(deBaat, 1999)	Operational processes	External Infrastructure (Suppliers, Partners, Customers) Business Architecture (Business Organisation, Business Processes, Business Information), Application Architecture (Presentation, Business Logic, Data Access), Technical Architecture (Middleware, Operating System, Hardware)
		Migration	Operations & Support,
		Infrastructure	Specification, Test, Training & Deployment
			Development & Maintenance
			Architecture & Engineering
GERAM	(IFIPIFAC, 1998)	Life-Cycle	Identification, Concept, Requirements, Design, Implementation, Operation, Decommission
		Genericity	Generic, Partial, Particular
		Views	Entity Model Contents, Entity Purpose, Entity Implementation, Entity Physical Manifestation
GRAAL	(van Eck et al., 2002)	Service Layers	Environment, Business mission and functions, Business processes, Software applications, Software platform, Processing and networking hardware
		Refinement	
		Lifecycle	Planning, Organizing, Directing, Controlling
		Aspects	Dictionary, Communication, Functions, Behavior, Quality
Herzum/Sims	(Herzum and Sims, 2000)		Functional, Application, Technical, Project Management

(*Continued on next page*)

Table 1 (*Continued*)

Framework	Source	Dimension	Values
Herzum/Sims	(Herzum and Sims, 2000)		Functional, Application, Technical, Project Management
IAF[1]	(Goedvolk and Rijsenbrij, 1999)	Main architecture areas	Business, Information, Information Systems. Technology Infrastructure
		Design phases	Contextual, Conceptual, Logical, Physical, Transformational
		Special viewpoints	Business and ICT System, Security. Governance
IFW	(Evernden, 1996)	Types of information	Organization (Strategy, Structure, Skills), Business (Data, Function, Workflow, Solution), Technical (Interface, Network, Platform)
		Levels of constraint	Deconstruction (Domain Concept, Domain Classification), Composition (Generic Template, Design Context), Implementation (Operational Bound)
		Content	Organisation Model, Financial Services Data Model, Financial Services Function Model, Financial Services Workflow Model, DesignWare, Finance Industry Solutions, Technical Model, Financial Application Architecture
		Transformation	
		Ownership	Global, Industry, Enterprise, Local, Individual
		Route maps	
MAD	(Meinema, 1999)		Inter-organizational, Organizational, Process, Information, Application, Distribution, Configuration
Maier/Rechtin	(Maier and Rechtin, 2002)		Data, Behaviour, Form, Purpose, Performance, Managerial
March	(Hermans, 2002)		Product, Process, Organisation, Information provisioning, Infrastructure Context, Concept, Logical Now, less than 2 years, 2 to 5 years
RM-ODP	(ISO/IEC CD 10746-1, 1994)		Enterprise, Informational, Computational, Engineering, Technology
Siemens	(Hofmeister et al., 1995, 2000)		Conceptual, Module, Execution, Code
Tapscott	(Tapscott and Caston, 1993)		Business, Work, Information, Technology Application,
TOGAF	(The Open Group, 2003)	Architecture Domains	Business, Data, Applications, Technology
		Architecture Continuum	Foundation, Common Systems, Industry, Organisation
Zachman	(Zachman, 1987) (Sowa and Zachman, 1992)	Types of description	Data, Function, Network, People, Time, Motivation
		Perspectives	Contextual, Conceptual, Logical, Physical, Implementation, Out-of-Context

[1] Recently IAF has included the "Enterprise" main architecture area, which comprises one holistic representation of the organization as a whole.

values into subvalues is shown in parentheses. The table will be the primary source of inspiration for our definition of dimension, and the base dimensions that we distinguish. Cells for dimensions and values that are empty indicate that the source does not explicitly name them.

Observations

Confusion

An analysis of existing frameworks and their dimensions leads to a number of observations:

- They use different terms for similar aspects, and similar terms for different aspects (for example: the term "business" in IFW is not the same as the term "business" in TOGAF).
- They often define terms only informally making it difficult to demarcate boundaries clearly (for example: where does the conceptual level end and the logical level begin?).
- They often do not name dimensions explicitly, leaving their interpretation up to the reader (an example is the March framework).
- They sometimes do not distinguish clear values within the dimensions, hindering effective communication (an

example is the Evernden Eight that leaves the exact content of all dimensions up to the reader).

- They often have slightly different sets of values for particular dimensions (see for example the IAF "design phases" dimension and the Zachman "perspectives" dimension).
- They sometimes have dimensions with values that do not have a clear relationship, which makes it hard to understand the dimension altogether (take for example the "special viewpoints" dimension in IAF).

These observations show that architecture frameworks are not the silver bullet for the confusion that exists when talking about architecture. Not only do individual frameworks leave us with some questions, but current architecture frameworks are also inconsistent with each other, making it necessary to tell someone which framework you use when talking about architecture. What is required in our view is an underlying concept of architectural dimension, but more on that later.

Essentials

Analysis of the frameworks also leads to another important observation, in that frameworks in essence are an attempt of the creator to enable clustering of architectural information in a way that suits a particular context and goal, with many parties involved. We see dimensions that are unnamed, which may express the lack of one overall concept. We perceive these unnamed dimensions as a struggle by the creator to capture in one stroke the main dimensions by which the various stakeholders structure their world.

We observe a division of dimensions into primary, secondary and supporting ones. The reason probably is that two dimensions are often enough to cover the required architectural descriptions. Also, on paper it seems most natural to represent the framework as a collection of cells, spread over the two dimensions of a matrix; one dimension is depicted horizontally (primary dimension), and another one vertically (secondary dimension). Sometimes other (supporting) dimensions are shown, or mentioned in the description of the framework.

The primary, horizontal dimension is often the type of information (topic), which can be divided into business and IT aspects. This distinction between business and IT is prominent in enterprise-class frameworks, but missing in application-class frameworks. This is perfectly explainable from the purpose of the framework: enterprise-class frameworks need to align business and IT, while application-class frameworks only need to model an IT solution. A general observation concerning this first dimension is that, although relationships exist, the values can be described fairly independent of one another.

The secondary, vertical dimension, in contrast, often is one that has a sequential aspect or is simply a partitioning in different levels of detail. With a sequential aspect there is a certain order in the construction of architectures that follows the values in this dimension. Examples of such dimensions are the IFW dimension "levels of constraint", and the IAF dimension "design phases". When devising such a dimension the framework creator must discover which architectural descriptions need to be fixed first, and which architectural descriptions need to be based on them. When the dimension is a partitioning into levels of detail, the higher rows contain a higher level of abstraction (fewer details) than lower levels. These two meanings of the secondary dimension (sequence versus levels of detail) are very similar since a design usually progresses from a high level of abstraction to a lower level of abstraction.

Dimensions are inherent in the paradigms people use, and prevailing paradigms can be a good source for concepts to build architectural dimensions from. Examples of these are the chain of control, the value chain, and the phases in development.

These observations might be helpful for those that want to describe their own architecture framework. They need to ask themselves: "what is the purpose of this framework?", "what are the types of information that need to be described?" and "what is the order in which we want to architect?". We believe that the best architecture framework is the one that provides answers that are most appropriate for a specific context.

Dimensions

We have used the term "dimension" informally several times in this paper already. It is an everyday word. Now we will try to formalize it. Using the resulting definition we will synthesize a list of base dimensions from existing frameworks. These base dimensions are further explained and illustrated subsequently.

Webster Online offers the following explanation of the word "dimension":

1a (1): measure in one direction; *specifically*: one of three coordinates determining a position in space or four coordinates determining a position in space and time (2): one of a group of properties whose number is necessary and sufficient to determine uniquely each element of a system of usually mathematical entities (as an aggregate of points in real or abstract space) <the surface of a sphere has two *dimensions*>; *also*: a parameter or coordinate variable assigned to such a property <the three *dimensions* of momentum> (3): the number of elements in a basis of a vector space **b**: the qualityof

Table 2 Proposed base dimensions

Dimension	Description
Type of information	The topic of the information (*business, organisation, technical)*
Scope	The extent of the information covered (*industry sector, organisation, domain, system family, system, component)*
Detail level	The amount of detail (*high, medium, low)*
Stakeholder	The target audience (*client, end-user, architect, analyst, developer)*
Transformation	The transformation phases that the architecture needs to cover (*current situation, short-term, medium-term, long-term)*
Quality attribute	The quality attribute that is being addressed (*functionality, reliability, usability, efficiency, maintainability, portability)*
Meta level	The amount of abstraction (*instance, model, meta-model, meta-meta-model, meta-meta-meta-model)*
Nature	The nature of the information (*policy, principle, guideline, description or standard)*
Representation	The way architectural information is represented (*formal, semi-formal, informal)*

spatial extension: MAGNITUDE, SIZE **c:** a lifelike or realistic quality **d**: the range over which or the degree to which something extends: SCOPE – usually used in plural **e**: one of the elements or factors making up a complete personality or entity: ASPECT

2 *obsolete*: bodily form or proportions

3 : any of the fundamental units (as of mass, length, or time) on which a derived unit is based; *also*: the power of such a unit

4 : wood or stone cut to pieces of specified size

5 : a level of existence or consciousness

With a little play of words from **1 a** (2) we like to see a dimension in the field of IT-architecture as an attribute of a piece of information which positions this piece of information in the total available information space. **1 e** shows that more than one dimension is needed to make up a complete architecture description. And **3** speaks of fundamental (base) units on which a derived (practical applicable) unit is based. So far the (serious) play of words. If we try to put it in one sentence, it would be something like:

An architectural dimension is a criterion to partition an architectural description into a set of segments, where each segment is identified by a unique value within a list of values associated with the dimension.

Architectural descriptions should document the dimensions used and the segments they cover in an introductory chapter. Standardizing these dimensions, their segments in particular, in a specific organizational context prevents semantic obscurities and introduces a shared architecture terminology.

Base dimensions

Based on our definition and existing architecture frameworks, we will now synthesize a list of nine base dimensions. The sources of inspiration for these dimensions are the existing architecture frameworks. We have studied the dimensions in these frameworks and transformed them into "pure" dimensions conforming to our definition. The resulting list is shown in Table 2, which includes a short description of the dimension and a hint at possible values. Since it is not our intent to standardize or formalize these values, they are just meant to illustrate the dimension. Also, we are not claiming that the set of nine dimensions is complete; other base dimensions may exist and could be added to our list.

Base dimensions in detail

We will now describe our proposed base dimensions in more detail. The first five are fairly common in architecture frameworks. The other four are used less frequently.

Type of information. This dimension is by far the most prevalent in architecture frameworks, and describes the subject of architectural information. Another way to look at this dimension is that it consists of the concepts that exist in domain-specific languages. At a high level this dimension can distinguish segments such as business, organisation, and technical. Within these segments a further segmentation typically exists. For example, IFW decomposes the technical segment into interface, network and platform segments. Some other frameworks that use this dimension are: 4+1, DYA, GEM, GRAAL, RM-ODP, Siemens, and TOGAF.

We perceive this dimension as a means to break down a complex situation into more or less independent aspects.

Together these aspects provide a conceptual model of the entire environment. Some frameworks are explicit about the relationship between aspects. An example is the GRAAL framework which claims a service provisioning sequence from "processing and networking hardware" to "software platform" to "software applications", and so further. TOGAF prescribes a design sequence from "business" to "information systems" to "technology", which we read as a claim that the business determines the information systems, and that information systems determine the technology.

Scope. This dimension describes the scope of the information covered. It is our proposal for a "clean" top down dimension, one that is easily understood. One way to decompose this dimension is with the values industry, organisation, organisational domain, system family, system, and system component. Scope is the main dimension of the Gartner framework (Rosser, 2002) with a different list of values. Different interpretations of the dimension are possible, interpretations that each may be valid from a specific point of view. The scope dimension is very much related to the ownership dimension in IFW, and it is implicitly used in the levels of constraint dimension in IFW. In particular, the design context and operational bound values in IFW have a system scope, while the upper levels have a domain scope.

Detail level. This dimension is based on the amount of detail, where levels with more information can be defined. A characteristic is that all information of the level above is kept, and that new information is added. The primary goal of varying the level of detail is to leave out those details that are not relevant or known in a particular context or at a particular moment in time. Since it is possible to add different types of detail, one could say that the detail level dimension comes in various types. Examples of frameworks in which we recognize a detail level dimension are: Zachman (perspectives dimension), IAF (design phases dimension), March (second unnamed dimension). We say "recognize" because the detail level portioning is a bit obscured by other meanings attached to these dimensions in the frameworks.

Stakeholder. This dimension uses the stakeholders that are addressed as primary criterion. Stakeholders are typically only interested in certain parts of the architecture. Defining descriptions for specific stakeholders was the intention of the Zachman perspectives dimension, but this also holds for other architecture frameworks such as 4+1, IAF and RM-ODP. Again, the pure meaning of the "stakeholder" dimension is obscured by other meanings attached to it in the frameworks.

Transformation The transformation dimension uses change in time as the criterion. It distinguishes the current situation from short-term, medium-term and long-term situations, including the transitions between them. A slightly different way to define this dimension is to not refer to specific moments in time, but to characteristics of the situation that can exist in time, like the levels in the Capability Maturity Model (CMM) initial, repeatable, defined, managed and optimised. Examples of frameworks that use this dimension are: Gartner, IFW, Evernden Eight and March.

Quality attribute. A number of dimensions in existing frameworks mention quality characteristics such as security, performance and usability, see for example IAF and Maier/Rechtin. In our view these characteristics can be considered as a separate dimension, with segments that each highlights certain quality characteristics. The values within this dimension are defined by quality frameworks. Various quality frameworks exist, such as the Extended ISO model (vanZeist et al., 1996). This dimension makes it possible to talk about, for example, a performance view or a security view. These last two views are also very common types of quality-driven views.

Meta level This dimension addresses those architectures that, instead of domain-specific models, provide general classifications and relationships. It really describes a metamodel; information about information. Consider for example a model that describes the types of components that may be developed, and the legal relationships between them. Multiple meta levels exist (meta-meta models, and so forth), but arguing that these are architectural in nature becomes increasingly difficult. The "meta level" dimension resembles the "detail level" dimension; the difference is that instead of less information meta-models describe different information. Evernden Eight is a framework that includes a meta level dimension.

Nature. This dimension determines the nature of the architectural information; is it a policy, principle, guideline, model or standard. Inherent in this dimension is the extent in which designers need to comply with the architectural information. A policy is clearly more important to follow than a guideline. The dimension is based on the dimensions as defined by Boar and DYA.

Representation This dimension uses the way to represent architectural information as criterion. One can choose between formal, semi-formal and informal representations. An informal representation is natural language, which leaves room for interpretation. Semi-formal means such as UML improve the welldefinedness. Formal description languages such as C2 and Rapide (Medvidovic and Taylor, 2000) are at the other extreme, but sometimes necessary to automatically generate models or reason about them. For example, a performance model based on Queueing Networks (Smith and Williams, 2002) provides a very accurate description of a system. The Evernden Eight framework also includes a representation dimension.

Usage

The list of base dimensions can be used in many different ways: as communication vehicle, checklist or basis for an architecture description or an architecture framework. The primary goal of the list is to facilitate communication about architecture in general. There are several ways to support this, such as documenting the values that an architectural description covers in the various dimensions in an introductory chapter. Also, in verbal communication these dimensions can be used to position an architectural description. Using the list as checklist allows one to check whether all relevant aspects have been taken into account for a specific architecture. Finally, the list can be used in the construction of a new architectural description or architecture framework. This means selection of the most applicable dimensions and values within those dimensions, and translating those to document structures.

Example

We will now exemplify the use of our list of base dimensions by positioning the view on architecture of the Rational Unified Process (RUP) (Rational Unified Process, 2002), an object-oriented software development method (see Table 3). The software architecture document (SAD), as RUP calls the architectural description, contains seven potential viewpoints that are inspired by the 4+1 model. In addition to the original viewpoints, also a data and user experience viewpoint are added. Looking at these viewpoints we see that they describe technical information about the system. RUP talks about the "software architecture" of a system, implying a system scope for the architecture. The detail level of the information is medium; it is not the intention of the SAD to be a detailed design. Looking at the activities that the SAD is input to, we derive that the target audience of the document are the designers and implementers of the system. The goal of the SAD is to be a short-term architecture; project members need to fully comply to it immediately. Although the emphasis of the SAD is on the functionality of the system, the impact on all other quality attributes also needs to be documented. The contents of the document are models of the system; no meta-models are described. Also, the nature of the architectural information is that it contains only models; no principles, guidelines or standards. The models are represented by UML models, which are supplemented with text.

Table 3 Positioning architecture within Rational Unified Process

Base dimension	Value
Type of information	Technical
Scope	System
Detail level	Medium
Stakeholder	Designer, Implementer
Transformation	Short-term
Quality attribute	All
Meta level	Model
Nature	Model
Representation	Text, UML diagram

IEEE 1471

The IEEE 1471 "recommended practice" defines concepts and their relationships that are relevant for architectural descriptions (IEEE Std 1471-2000, 2000). It also provides guidance on the structure of architectural descriptions. The main concepts standardised are "architecture", "architectural description", "concern", "stakeholder", "viewpoint" and "view", see Fig. 3. Architecture is defined 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". Architectural descriptions are segmented into views. A view addresses a related set of stakeholder concerns and is constructed in accordance with the specification that is laid down in a viewpoint. Together the views address all the concerns of the stakeholders.

Since its publication in 2000, IEEE 1471 has received much appraisal. The concepts of stakeholders, concerns and views are accepted as essential. The terminology proposed by

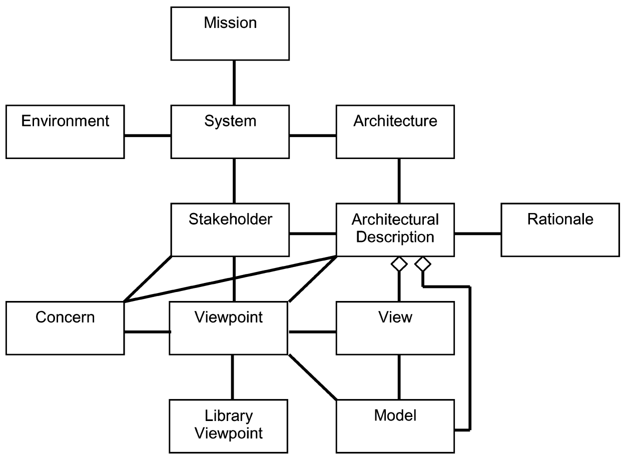


Fig. 3 IEEE/ANSI Std 1471 conceptual model

IEEE 1471 is now being used by many architects. The focus on concerns of stakeholders is a good stimulus for otherwise possibly too technically oriented IT architects. After all, it is the interests of the stakeholders that need to be served.

Although IEEE 1471 is an important contribution to standardising architecture terminology, it still leaves a number of things unspecified. Most importantly, IEEE 1471 does not propose nor prescribe any specific viewpoint, which might confuse architects and stakeholders. In a specific context, two architects can easily disagree on who the stakeholders and their concerns are, and what information is needed to address these concerns. Also, if a view contains a lot of architectural information, it needs to be structured, bringing back the needs for which frameworks have been defined. This also holds at the enterprise level, where many IEEE 1471 compliant architectural descriptions may need to be made accessible. So, even if all IT architects would follow IEEE 1471, architectural information could still be very different up to a point where documents are still not accessible, nor comparable. The "dimension" concept provides a means to further structure IEEE 1471 views into more manageable chunks.

We don't write this in critique of IEEE 1471, but we do feel compelled to raise some arguments against a view of IEEE 1471 as the silver bullet where it comes to architectural descriptions. Also with IEEE 1471 at hand, there still is a need for additional support to help communication about IT architecture.

We also see a mismatch between IEEE 1471 and existing architecture practice as represented by the frameworks in our overview. IEEE 1471 requires a view to address a set of related concerns. The "chunks" in which existing frameworks divide the architectural information are addressing many concerns, but it is not obvious these concerns are "related" in the sense of IEEE 1471. Our guess is they aren't, but a difficulty here is that IEEE 1471 does not specify what "related" exactly means.

Conclusions and future work

There are many differences between existing architecture frameworks. Partly this can be explained from their original goal, and the context from which they originated. A commonality is that architectural information is often organised in a matrix that is bound by two dimensions: one dimension typically addresses the type of information, and a second one having a sequential order.

In this paper, we propose the use of nine base dimensions: Type of information, Scope, Detail level, Stakeholder, Transformation, Quality attribute, Meta level, Nature and Representation. These base dimensions allow us to better understand and compare existing frameworks, or to create a new framework. They also ease the understanding and communication of architectural descriptions.

There still remains a lot of work to be done in architecture description standardization. In particular, the values within the dimensions described need to be widely agreed upon. This will lead to standardized architectural viewpoints (library viewpoints), and will ultimately contribute to the further maturation of the architect profession. We would like to understand more of the circumstances in which the different frameworks function.

We recommend the use of IEEE 1471 and would like to see more constructive debate to come to effective application of this standard.

Acknowledgments We want to thank everyone that has provided us with helpful comments and feedback. In particular, we thank Philippe Spaas, Harry Hendrickx, Jan Campschroer, Erik Vermeulen, and Michiel Perdeck for their review comments. We also want to thank the anonymous reviewers of an earlier version of this paper.

References

Boar BH. *Constructing Blueprints for Enterprise IT Architecture*. Wiley, 1998.

de Baat JM. CMG's Multi-Channel Management Vision on Architecture, 1999.

CIMOSA—Open system architecture for CIM. ESPRIT Consortium AMICE, Springer-Verlag, Berlin, 1993, (ISBN 3-540-56256-7), (ISBN 0-387-56256-7).

Evernden R. The information framework. *IBM Systems Journal*, 1996.

Evernden R. Evernden eight. 4th, resource 2002.

GERAM: Generalised enterprise reference architecture and methodology. IFIPIFAC Task Force, Version 1.6.2, 1998.

Goedvolk H, Rijsenbrij D. White paper integrated architecture framework. version 1.0, 1999.

Hermans L. Uitbuiten synergie ICT- en business strategie. Informatie, ten Hagen Stam, 2002.

Herzum P, Sims O. *Business Component Factory*. John Wiley & Sons, 2000.

Hofmeister C, Nord RL, Soni D. An industrial perspective of software architecture. In: *Proceedings of the Eleventh International Conference on Data Engineering*, IEEE Computer Society, Taipei, Taiwan, March 1995.

Hofmeister C, Nord R, Soni D. *Applied Software Architecture*. Addison Wesley, 2000.

IEEE Std 1471-2000. IEEE recommended practice for architectural description of software-intensive systems, 2000.

ISO/IEC CD 10746-1, Basic reference model of open distributed processing, 1994.

Kruchten P. The $4 + 1$ view model of architecture. *IEEE Software*, 1995.

Lassing N, Rijsenbrij D, van Vliet H.Viewpoints on modifiability. *International Journal of Software Engineering and Knowledge Engineering* 2001;11(4):453-478.

Maier MW, Rechtin E. *The Art of Systems Architecting*. CRC Press, 2002.

Medvidovic N, Taylor RN. A classification and comparison framework for software architecture description languages. *IEEE Transactions on Software Engineering* 2000;26(1):70–93.

Meinema JL.*Corporate Architecture: A Conceptual Approach*. University of Twente, 1999.

Rational Unified Process. Rational corporation, Version 2002.05.20. 005, 2002.

Rosser B. Defining architecture for IT: A framework of frameworks. Gartner, Research Note, 12, 2002.

Scheer AW. *Architecture of Integrated Information Systems*. Springer, 1992.

Smith C, Williams L. *Performance Solutions*. Addison Wesley, 2002.

Sowa JF, Zachman JA. Extending and formalizing the framework for information systems architecture. *IBM Systems Journal* 1992;31 (3).

Tapscott D, Caston D. Paradigm Shift—*The New Promise of Information Technology*. McGraw-Hill, 1993.

The Open Group: The Open Group Architectural Framework, Version 8.1 2003.

van Eck PAT, Blanken H, Fokkinga M, Grefen PWG, Wieringa RJ. A conceptual framework for architecture alignment guidelines—Project GRAAL WP1 Whitepaper, Department of Computer Science, University of Twente, 17, 2002.

Wagter R, et al. DYA: Snelheid en samenhang in business en ICT-architecture. Tutein Nolthenius, 2001.

Youngs R, et al. A standard for architecture description. *IBM Systems Journal* 1999;38(1).

Zachman JA. A framework for information systems architecture. *IBM Systems Journal* 1987;26(3).

van Zeist B, et al. *Kwaliteit van Software Producten, Praktijkervaring met een Kwaliteitsmodel*. Kluwer Bedrijfsinformatie, 1996.

Danny Greefhorst is a Senior IT Architect at IBM Business Consulting Services. His expertise lies in the area of enterprise application architecture, integration and development. Before joining IBM he worked as senior researcher for the Software Engineering Research Center. Danny has a M.Sc. in Computer Science from the University of Utrecht.

Henk Koning has over 20 years practical experience in system development and architecture. This article is part of his Ph.D. study on "Communication of IT-architecture".

Hans van Vliet is a Professor in Software Engineering at the Vrije Universiteit. His research interests include software architecture and software measurement. Before joining the Vrije Universiteit, he worked as a researcher at the Centrum voor Wiskunde en Informatica (Amsterdam) and he spent a year as a visiting researcher at the IBM Almaden Research Center in San Jose, California. Hans has an M.Sc. in Computer Science from the Vrije Universiteit and a Ph.D. in Computer Science form the University of Amsterdam.