11. An Ontological Approach to Characterising Enterprise Architecture Frameworks

Oddrun Pauline Ohren
SINTEF, Norway. Email:[oddrun.ohren@sintef.no]

Currently, several enterprise architecture frameworks exist, and there is a need to be able to communicate about them. To this end we have proposed an Architecture Framework Ontology (AFO) providing characteristics to be assigned to the framework under consideration. AFO is then used to characterise and compare six existing frameworks, and results from this task are presented.

1. INTRODUCTION

During the last couple of decades, quite a few enterprise architecture frameworks have been presented. An architecture framework (AF) may be viewed as *a set of rules, guidelines and patterns* for describing the *architecture* of *systems*. Traditionally, in this context architecture means *information system* architecture or *software* architecture. However, a growing understanding of the importance of the context within which an application is to operate, gave rise to a "new" type of architecture, namely enterprise architecture and with it enterprise AFs. While such frameworks differ considerably in a number of respects, their application domains tend to overlap. The fact that there exist several potentially useful frameworks for any architecture effort creates the need for users to be able to assess frameworks and compare them with each other. Given their relative comprehensiveness and complexity, this is no easy task.

Although some work is performed concerning architecture framework issues (Mili, Fayad *et al.*, 2002; Martin and Robertson, 2003), we have not been able to find a simple framework for assessment and comparison of AFs.

As a first step towards such a mechanism, this paper outlines an ontology for characterising AFs. (A short version of this paper (Ohren, 2004) is presented in the poster session at the EMOI- INTEROP 2004 Workshop).

Such an ontology provides a *vocabulary*, a conceptualisation for communicating about architecture and AFs. It also supports *comparison* between AFs, as it points to distinguishing features of AFs.

Also, when embarking on an architecture project it is important to choose an AF that fits the task at hand. Although the proposed ontology is not primarily directed towards matching problems with AF, it does identify the characteristic features of a framework, which should help evaluating its suitability for the case at hand.

1.1 Related work

While some interesting work on AF issues has been published, this is as yet not a very developed research area, especially when it comes to enterprise AFs. (Dobrica and Niemelä, 2002) and (Medvidovic and Taylor, 2000) study two different aspects of *software* architecture, namely architecture analysis methods and architecture description languages, each proposing a framework for classification and comparison of analysis methods and description languages, respectively. As for enterprise architecture, there exists a few studies related to specific frameworks, for example (Cook, Kasser *et al.*, 2000) assessing the defence-oriented C4ISR[69] AF and (Goethals, 2003), studying the architecture products prescribed in several frameworks. (Martin and Robertson, 2003) performs an in-depth comparison of two distinct enterprise framework types, whereas (Mili, Fayad *et al.*, 2002) identifies some important issues related to the *management* of enterprise AF in general.

2. CHARACTERISING ARCHITECTURE FRAMEWORKS

In this chapter we propose a conceptualisation for talking about AFs, evaluating them and relating them to each other.

2.1 Ontological basis

A study of six enterprise AFs (Ohren, 2003) forms the basis for the ontology. These frameworks are: Federal Enterprise Architecture Framework (FEAF) (Chief Information Officers (CIO) Council, 1999), Department of Defense Architecture Framework (DoD AF) (Department of Defense Architecture Framework Working Group, 2003), Treasury Enterprise Architecture Framework (TEAF) (Department of the Treasury CIO Council, 2000), Zachman Framework (ZIFA), The Open Group Architectural Framework (TOGAF) (The Open Group, 2002) and Generalised Enterprise Reference Architecture and Methodology (GERAM) (IFIP-IFAC, 2001).

The proposed ontology has also been influenced by the ongoing work on MAF (Aagedal, Ohren *et al.*, 2003). MAF is a high-level *model-based* AF, implying a strong focus on models as the main formalism for describing architectures. The MAF metamodel (the model of which MAF is an instance) has greatly inspired the identification of the distinguishing features forming the framework ontology presented here.

2.2 Designing the Architecture framework ontology (AFO)

The framework ontology is realised as a class hierarchy with `Architecture framework characteristic` as the top node. The leaf nodes are instantiated, forming a set of concrete characteristics to be applied to the AF under consideration.

The framework ontology was designed according to the following guidelines:

- The concepts should *characterise well*, that is, refer to features that are *important* in an AF, i.e. important for deciding if and how to utilise the framework in a specific situation.

---

[69] Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance

- The concepts should refer to features that *discriminate* well between the AFs. However, this consideration may be over-shadowed by the *importance* requirement.
- The concepts should be clearly and thoroughly defined, with particular regard to revealing implicit similarities between the frameworks, especially in cases where the similarity is obscured by diversity in vocabulary.

The framework ontology consists of the following top level concepts, each representing a type of characteristic of AFs: `Application domain characteristic`, `Ontological characteristic`, `Prescription for ADs`, `Methodological characteristic`, and `Relation to other frameworks`.

Table 1 lists the classes of AFO in the left column, representing *types of* characteristics of AFs. Subclasses are indented. The corresponding instances are listed to the right, representing *concrete* characteristics to be assigned to the AFs.

Table 1 The classes and instances of AFO

Concept	**Instance (individual characteristic)**
Application domain characteristic	
System scope	System type in general, US Department of Defense, US Department of Treasury, US Federal Government
System type	Enterprise, Software system
Ontological characteristic	
Ontology scope	Application domain, Architecture
Ontology form	Formal concept model, Glossary of terms
Prescription for ADs	
Prescription regarding AD content	Enumeration of products, Implicit specification of products
Prescription regarding AD organisation	Architecture domain, Analytical approach,, Life-cycle phase, Stakeholder, Level of system abstraction
Prescription regarding AD representation	Formal representation, Informal representation
Methodological characteristic	AD Development process, Architecture evolution support, Principles of conformance and consistency
Tool support characteristics	Full tool support, No tool support, Some tool support.
Relation to other framework	Developed from, May be combined with, Used in, Uses

Note that AFO should not be viewed as a fixed and complete classification scheme for AFs. The currently defined instances mainly have their origin from the AFs in the study, and do not form a complete set. However, introducing additional characteristics in the ontology is not a problem; there is no requirement of the instances being semantically disjunctive. Nor does AFO require the set of instances

of a class to span the class. On the other hand, defined associations between classes make it possible to relate instances to each other in various ways. For example, instances of the characteristic type System scope may be internally related by the included in relation. We do not require that a complete and disjunctive set of scopes be chosen as instances. Instead, the user is free to introduce a scope that fits the AF at hand. The important thing is that *we know it is a scope.*

2.3 The concepts of AFO

In this section we will look into the main concepts of AFO in more detail.

The application domain of an architecture framework

According to the IEEE 1471 standard (IEEE, 2000) 'architecture' is the 'fundamental organization of a *system* embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution'. The term 'system' is defined as 'a collection of components organized to accomplish a specific function or set of functions'. The term 'system' being extremely generic, its range of instances might be expected to be somewhat heterogeneous. Therefore, when evaluating an AF, it is crucial to know what kind of systems it is intended to serve. In AFO this is represented by the concepts Application domain characteristic, with subtypes System type and System scope. Examples of system types are *software system* and *enterprise*. System scope is a feature intended to restrict the System type, and may be various things like geographical scope (e.g. Norwegian), industrial branch (e.g. chemical industry) and others.

AFs as providers of conceptualisations of architecture

Most AFs use their own more or less proprietary vocabulary when describing architectures. Our study of the frameworks mentioned above shows that there is a tendency to supply commonly used words with very specific semantics. Moreover, different frameworks often use the same word in different meanings. This fact tends to obscure both differences and similarities between frameworks, especially if the frameworks do not provide explicit definitions of their key terms.

The Ontological characteristic indicates whether the framework provides ontologies for the architecture domain and relevant aspects of the application domain, and possibly other relevant areas. Two subtypes are defined for this characteristic, Ontology scope and Ontology form. Ontology scope indicates the domains covered by the ontology (e.g. architecture), whereas the Ontology form indicates how the ontology is realised in the framework.

AFs as templates for architecture descriptions

Most AFs contain prescriptions concerning the architecture description as artifact. While their degree of specificity varies considerably, these prescriptions usually deal with what information should be included in the architecture description, how it should be structured and sometimes how it should be represented.

The Framework ontology includes four concepts to this end; Prescription for ADs and its subtypes Prescription regarding AD content, Prescription regarding AD organisation, and Prescription regarding AD representation.

AFO presently contains four instances of Prescription regarding AD organisation, each of which constituting a structuring criterion for the AD products. These are:

- Architecture domain: In the context of enterprises, it is common to recognise three or four types of architecture, each corresponding to its particular domain: Business architecture, Information system architecture (often subdivided into Data architecture and Applications architecture) and Technology architecture. Note: Architecture domain as a structuring criterion for a collection of architecture products should not be confused with the application domain of the framework as such.
- Analytical approach: Analysing a system often implies focusing on one angle at a time. This criterion orders the architecture products according to the angles applied in the various analyses producing the products. Typical examples of analytical approaches are: Functional analysis (how does the system operate?), Structural analysis (which components do the system consist of, and how are they structured?), Spatial analysis (at which locations does the system reside, how is it distributed?) and Information analysis (what information is handled in the system?).
- Stakeholder: According to IEEE 1471, any system has one or more stakeholders, each of which has certain interests in that system. In cases where stakeholders are defined solely in terms of another criterion, the two criteria will result in the same logical partitioning, although the level of granularity may differ.
- Life-cycle phase: Organises the architecture products according to the life-cycle phase (e.g. design phase) addressed by the product.
- Level of system abstraction: Organises the architecture product according to the level of abstraction at which the system in question is described in the product. Examples of system abstractions are: *Physical manifestation* of the system, *Implementation* of the system and Purpose of the system.

AFs and architecture development methodology

Describing an existing or future architecture is often a major undertaking, and it is by no means obvious how to approach such a task, even if one uses an AF which details the content of the resulting description. *Whether* and eventually *how* an AF supports the architecture description development process is an important feature of the framework.

To indicate the possible methodological support offered by a framework, the concept Methodological characteristic with subtype Tool support characteristic are used. Examples (instances) of Methodological characteristic are whether the framework specifies an AD development process, or provides any Architecture evolution support, whether it is supported by software tools, etc. Examples of Tool support characteristic are No tool support and Full tool support.

Relations between architecture frameworks

Some frameworks are related to each other in various ways. Whether factual (declared) or purely conceptual, some of these relations reveal important information about the background of the participating frameworks, in which case they should be identified and documented. Examples of instances are Used in between two

frameworks of which one is used as a part of the other and `Developed from` between frameworks where one is part of the history of the other.

3. COMPARING SIX ENTERPRISE ARCHITECTURE FRAMEWORKS

In this chapter AFO is used to characterise the six frameworks from the study. For reasons of space, we focus on a subset of AFO, discussing only the characteristic types believed to be the most interesting or challenging. For each chosen characteristic class, all six frameworks will be assigned one or more of its instances (concrete characteristics). The findings are summarized in Table 2.

3.1 Application domain

System type and System scope

All frameworks in the study claim to be *enterprise* AFs. For all except GERAM, the interpretation of 'enterprise architecture' is 'architecture of information systems in an enterprise context' rather that 'architecture of the enterprise as such'. However, all the frameworks prescribe inclusion of key operational and organisational information; hence it is fair to assign `Enterprise` as `System type` to all of them.

As for System scope, three of the AFs are designed by and for the US Government, hence their system scope must be specified accordingly: The scope of FEAF is the `US Federal Government` in general, while DoD AF and TEAF have narrower scopes; `US Department of Defense` and `US Department of Treasury`, respectively.

The frameworks TOGAF, GERAM and Zachman do not specify any restrictions regarding scope, hence they are applicable to any system of the type given by their `System type` characteristic, meaning they will be characterised by the `System scope` instance called `System type in general`.

AFO Shortcomings

Even though FEAF, DoD AF and TEAF are designed for use within the US Government, this might be more of a *declared* restriction than a *factual* one. For example, FEAF has few, if any, features that makes it unusable in other enterprises than the US Government. The distinction between declared and factual delimitations of the application domain is not as yet addressed in AFO.

FEAF aims to provide support for developing Federal wide and multi-departmental architectures, as well as a high level structure with which to link more specific architecture efforts within a single department. This information is not easily expressed with AFO, although part of it may be conveyed by using the association `included in` defined between system scopes. Thus the scopes of DoD AF and TEAF may be modelled as parts of the scope of FEAF.

3.2 Prescriptions about the architecture descriptions

Prescriptions regarding AD content

Within the six frameworks studied, there are great variations concerning the level of detail and abstraction at which the frameworks specify the content of an architecture description.

The most specific, thorough and formal frameworks in this respect are DoD AF and TEAF, both enumerating specific architecture products to be included in the architecture description, along with prescriptions for data representation. TEAF has derived its list of products from the Department of Defense's C4ISR AF (C4ISR Architecture Working Group, 1997) (not part of our study). Moreover, DoD AF is generally based on C4ISR, hence the product collections specified by TEAF and DoD AF turn out to be very similar, although TEAF naturally has had to adapt the products for use by Treasury.

FEAF also enumerates a list of architecture products to be developed, although this is less comprehensive and far less formal.

The three remaining frameworks, TOGAF, GERAM and Zachman are, as we have seen, more generic regarding application domain (`System scope` was set to `System type in general`), and are naturally less specific when listing architecture products. However, all of them specify in more or less general terms the *types* of deliverables or products that should be developed to form the architecture description. A further distinction exists between TOGAF and the others. TOGAF focuses more or less exclusively on methodology, and its specification of architecture products is to be considered more as an example than a prescription.

Prescription regarding AD organisation

With the exception of TOGAF, all the frameworks in the study employ some kind of structuring principle(s) upon the AD. For AFs offering methodological guidance, such criteria do not merely organise the architecture products, but are usually also reflected in the development process. The number of structuring principles prescribed in a single framework varies from 1 to 3. The result of applying them is a *partitioning* of the set of architecture products into subsets, each of which forming a logical unit expressing a *view* on the system in question. The purpose of splitting up the collection of architecture products is invariably to simplify and adapt the AD description to individual user groups, while internally maintaining an integrated, consistent and complete AD. Thus, capturing and visualising the whole architecture without getting lost in details is made possible.

Zachman organises the architecture products in a matrix where the rows are defined by a specific set of *stakeholders* (Planner, Owner, Designer, Builder, and Subcontractor), and the columns are called *dimensions* obtained by applying interrogatives (what, who, where...) when eliciting information about the system. Each stakeholder is defined in terms of his interest in an architecture domain, hence the AFO terms `Stakeholder` and `Architecture domain` are assigned to Zachman, even though the two characteristics are redundant. Zachman's *dimension* concept corresponds to the AFO term `Analytical approach`, hence this term is added to the characterisation of Zachman's structuring principles.

FEAF defines four levels at which to view the architecture. Going from level 1 to 4 implies adding detail for each level. The metaphor used to illustrate this is the observer's distance from the system. The closer one gets, the more details are visible. Note that this model does not impose an organisation of the architecture description, but a kind of zoom effect, presently not possible to represent by AFO. The actual description products are to be found at level 4. Here an adaptation of Zachman is used to organise the architecture products. The rows are still stakeholders, and the

columns, although labelled as architecture domains (data, applications, technology), by inspection have more in common with `Analytical approach` than with `Architecture domain`. Hence, the structuring of architecture products in FEAF may be characterised by `Stakeholder`, `Architecture domain` and `Analytical approach`.

TEAF also bases its organisation of architecture products on Zachman, but has simplified it to four rows and four columns. However, the rows (in TEAF called *perspective*) still represent stakeholders defined like those in Zachman and FEAF, hence should be characterised by `Stakeholder` and `Architecture domain`. The columns (in TEAF named Functional, Information, Organisational and Infrastructure *views*) seem best characterised by `Analysis approach`, although the Infrastructure column (as opposed to the Technology architecture column of FEAF) is technology oriented even at the business level, and as such may be said to cover the technology domain. This represents a deviation from the original Zachman matrix, in which the technology domain is covered by the Builder's row rather than the 'where' column.

DoD AF operates with a fixed partition of three elements called *views* (Operational, System and Technical views) as the sole organising principle. The documentation of the views implies a close correspondence to the `Architecture domain` criterion, the DoD AF views covering the Business, Information system and the Technology domains, respectively. A comparison with TEAF, which prescribes the same products as DoD AF, confirms this. DoD AF's Operational view corresponds to the perspective of Planner and Owner (Business domain) in the TEAF matrix, while the System view roughly corresponds to the perspective of Designer and Builder. Technical view corresponds to the Infrastructure column of the Planner and Owner perspective in TEAF, hence covering the technology domain (see discussion on TEAF in the paragraph above).

GERAM uses a three-dimensional structuring principle, consisting of life-cycle phase dimension, genericity dimension (generic, partial and specific), and a third called view dimension (content view, purpose view, implementation view and physical manifestation view). Although the view dimension has similarities to the Stakeholder dimension in the Zachman varieties, we chose to assign `Life-cycle phase` and `Level of system abstraction` as the organising characteristics of GERAM. So far, the genericity dimension can not be expressed in AFO.

TOGAF does not prescribe any specific structuring of architecture products. However, TOGAF ADM (methodology part) is partly organised according to architecture domains (first build business models, then information system models, etc).

Methodological characteristics

In general, the frameworks in our study focus on the specification of the architecture products rather than how to generate them. However, it is equally true that most of them do offer *something* in the way of a method or approach.

FEAF offers guidelines for architecture development in a separate document, in which an eight step development process is outlined. Also, FEAF talks explicitly about *as-is* architecture, *target* architecture and *transitional* processes (from as-is to target), hence architecture life-cycle issues are addressed. FEAF is therefore to be

characterised by the AFO instances AD Development process and Architecture evolution support.

TEAF does not include an AD development process, but encourages each bureau to supplement TEAF with a development methodology suited to its need. As for architecture life-cycle issues, TEAF does address things like Enterprise Architecture Roadmap and Enterprise Transition Strategy. TEAF is therefore to be characterised by the AFO instance Architecture evolution support.

Table 2 Summary of key findings

Characteristic	**FEAF**	**DoD AF**	**TEAF**	**Zachman**	**TOGAF**	**GERAM**
Prescription for AD content						
Explicit enumeration of products	X	X	X			
Implicit specification				X	X	X
Prescriptions for AD organisation						
Arch.itecture domain	X	X	X	X	(X)	
Analytical approach	X		X	X		
Life-cycle phase						X
Stakeholder	X		X	X		
Level of system abstraction						X
Methodological characteristics						
AD development process	X	X		X	X	
Architecture evolution support	X		X		X	
Relations to other frameworks						
Uses	Zach.		Zach. C4ISR			
Used in				FEAF TEAF		
Developed from		C4ISR				
May be combined with					any	

DoD AF offers guidance for architecture development in terms of a briefly described six step development process. Although DoD AF mentions architecture life cycle in the latest version, it is still pretty much a framework for a snapshot architecture. DoD AF is therefore to be characterised by the AFO instance AD Development process.

Zachman is a commercial product around which a number of services are offered, including architecture development guidance in the form of seminars. Zachman is therefore to be characterised by the AFO instance AD Development process

In many respects GERAM is more generic than the other frameworks. For example, its guidance concerning methodology is on a meta level compared to the others, as GERAM is mainly concerned with giving directions for methodology

development. Hence, it is not evident how the existing characteristics in AFO can be assigned to GERAM and at the same time conveying the difference in abstraction level.

TOGAF has methodology as its main concern, and is definitely to be characterised by `AD Development process` and `Architecture evolution support`.

Relations between frameworks

Our study of the six AFs has indicated that several of them are related, for example by sharing each other's history. Both TEAF and DoD AF have derived their set of architecture products from C4ISR AF. Adaptations of Zachman form a major part of FEAF as well as TEAF. TEAF proclaims to be compliant with FEAF. TOGAF (ADM, the methodology part) may be used in combination with any of the others. The exact assignments of AFO terms to represent this are shown in Table 2.

4. CONCLUSION AND FURTHER WORK

Given the increasing inclination to cooperate both across enterprises and across borders within an enterprise on one hand, and the number of different enterprise AFs on the other, we have in this article argued for the need to be able to assess, compare and generally communicate about AFs as artifacts separate from the architecture descriptions that they produce. To this end we have proposed an Architecture Framework Ontology (AFO) providing *characteristics* to be assigned to the framework under consideration. AFO has been used to characterise and compare six existing frameworks. Performing this task has revealed several shortcomings in the specific AFO as well as the general approach, and as such contributed to the future research agenda.

AFO is mainly focused on identifying distinguishing features of AFs. However, while it is useful to be able to compare AFs in a systematic way, there is also a need to perform *assessment* of frameworks, e.g. evaluate the suitability of a particular framework to the problem at hand. One way of obtaining this is to extend AFO with concepts related to architectural problem characterisation.

To support interoperability *within* and *between* enterprises, we need to be able to *relate architecture descriptions* created by different frameworks, preferably with some kind of computer support, which suggests a model-based approach to architecture descriptions. In a model-based world relating *descriptions* means relating *models*, which again requires a consistent mapping between their metamodels. This is, modestly phrased, a non-trivial task, and should get a lot of attention in the years to come.

Acknowledgments

This work is sponsored by the Norwegian Defence Logistics Organisation.

REFERENCES

Aagedal, J. Ø., O. Ohren, *et al.* (2003). Model-Based Architecture Framework, v 3.0: SINTEF Telecom and Informatics. 68.

C4ISR Architecture Working Group (1997). C4ISR Architecture Framework, v. 2.0: US Department of Defense.

Chief Information Officers (CIO) Council (1999). Federal Enterprise Architecture Framework. V. 1.1. [Washington DC, USA]: Chief Information Officers Council (CIO Council).

Cook, S., J. Kasser, *et al.* (2000). Assessing the C4ISR Architecture Framework for the Military Enterprise. International Command and Control Research and Technology Symposium, 5. 24-26 October 2000, Canberra, Australia.

Department of Defense Architecture Framework Working Group (2003). DoD Architecture Framework. Vol. I-II; Deskbook. Available from: http://aitc.aitcnet.org/dodfw/

Department of the Treasury CIO Council (2000). Treasury Enterprise Architecture Framework. V.1. [Washington DC, USA]: Department of the Treasury.

Dobrica, L. and E. Niemelä (2002). "A survey on software architecture analysis methods." IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 28(7): 638-653.

Goethals, F. (2003). An Overview of Enterprise Architecture Framework Deliverables; A study of existing literature on 'architectures'. http://www.econ.kuleuven.ac.be/leerstoel/SAP/downloads/Goethals%20Overview%20existing%20frameworks.pdf

IEEE (2000). IEEE Std 1471-2000: IEEE Recommended Practice for Architectural Description of Software-Intensive Systems: IEEE,.

IFIP-IFAC (2001). GERAM: Generalized Enterprise Architecture Architecture and Methodology. ISO 15704:2000. Industrial automation systems; Requirements for enterprise reference architecture and methodologies. Annex A. GERAM. T. C. 184: ISO.

Martin, R. and E. Robertson (2003). A comparison of frameworks for enterprise architecture modeling. Conceptual Modeling - Er 2003, Proceedings. Berlin: SPRINGER-VERLAG BERLIN. 2813. 562-564.

Medvidovic, N. and R. Taylor (2000). "A classification and comparison framework for software architecture description languages." IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 26(2000)(1): 70-93.

Mili, H., M. Fayad, *et al.* (2002). "Enterprise frameworks: issues and research directions." Software-Practice & Experience 32(8): 801-831.

Ohren, O. P. (2003). Rammeverk for beskrivelse av virksomhetsarkitektur [in Norwegian]. Oslo: SINTEF. 61.

Ohren, O. P. (2004). Ontology for characterising architecture frameworks. To be presented at the INTEROP Workshop "Enterprise modelling and ontologies for interoperability" EMOI - INTEROP., Riga, Latvia.

The Open Group (2002). The Open Group arcitectural framework (TOGAF), version 8. "Enterprise edition". Reading, UK and San Francisco, USA: The Open Group. Available on the web at http://www.opengroup.org/architecture/togaf8/.

ZIFA The Zachman Institute for Framework Advancement. http://www.zifa.com