EAF² – A Framework for Categorizing Enterprise Architecture Frameworks

Ulrik Franke, David Höök, Johan König, Robert Lagerström, Per Närman, Johan Ullberg, Pia Gustafsson, and Mathias Ekstedt

Industrial Information and Control Systems
Royal Institute of Technology
100 44 Stockholm, Sweden
{ulrikf, davidh, johank, robertl, pern, johanu, piag, mek101}@ics.kth.se

Abstract—What constitutes an enterprise architecture framework is a contested subject. The contents of present enterprise architecture frameworks thus differ substantially. This paper aims to alleviate the confusion regarding which framework contains what by proposing a meta framework for enterprise architecture frameworks. By using this meta framework, decision makers are able to express their requirements on what their enterprise architecture framework must contain and also to evaluate whether the existing frameworks meet these requirements. An example classification of common EA frameworks illustrates the approach.

Keywords: Enterprise Architecture, Enterprise Architecture Framework, Classification of frameworks.

I. INTRODUCTION

During the last decade, Enterprise Architecture (EA) has grown into an established approach for management of information systems in organizations. EA is model-based, in the sense that diagrammatic descriptions of the systems and their environment constitute the core of the approach. EA models increase the general understanding of an organization’s business and information system landscape and aids in decision making.

Within the practice of EA, a framework can be considered a conceptual structure of what an EA should contain and how to create it, i.e. a set of models, principles, approaches, standards and visualizations that guide the development of enterprise architectures [1]. Following the 1987 introduction of the Zachman framework for information systems architecture [1], several EA frameworks have been introduced, including The Open Group Architecture Framework (TOGAF) [2], the Department of Defense architecture framework (DoDAF) [3], the Federal Enterprise Architecture (FEA) [4], and the CIM Open System Architecture (CIMOSA) [5].

These are all EA frameworks, but they differ in content and target audience. While TOGAF details the process of creating an EA but is less detailed on actual modeling, DoDAF emphasizes models and metamodels. There is, however, no consensus on the proper contents of EA frameworks. We propose a method to characterize the different EA frameworks and match them with the goals and needs of different users. This paper proposes a meta framework, called the EA Framework (EAF²) for this classifying task. At least three distinct audiences can be discerned for use of the EAF². Firstly, CIO:es and other IT decision makers can benefit from a meta framework that enables them to select an appropriate EA framework for their specific purposes. Thus, someone looking for an architecture maintenance process might choose TOGAF, whereas someone looking for a metamodel might instead choose Archimate. Secondly, EA framework developers and standardization committees can look into the EAF² for directions on how to further develop frameworks to support distinct practitioner needs. Thirdly, in an academic context, the present contribution is an incremental step towards a unified idea of what constitutes an EA framework.

The rest of this paper is structured as follows. Section 2 describes some related work on framework classification. Section 3 outlines the construction of the EAF², while section 4 details the framework itself. Section 5 illustrates the use of the framework, and section 6 concludes the paper.

II. RELATED WORK

The most famous related work for identifying general components of enterprise architecture frameworks is the Generalized Enterprise Reference Architecture and Methodology (GERAM) [6]. GERAM was created by the IFIP/IFAC Task Force from existing frameworks. By identifying overlaps in the frameworks CIMOSA, GRAI/GIM and PERA [7], the IFIP/IFAC Task Force created a general reference architecture, intended to organize existing enterprise integration knowledge. Since the birth of GERAM, however, a vast variety of frameworks has evolved. Work on mapping frameworks onto GERAM for identifying conformity with the general components is presented in [8] and [9]. These sources both indicate that GERAM is a useful baseline for EA and enterprise engineering, but they contend that GERAM does not provide an eternal unified generalization able to stand the test of time. To be able to represent and classify present state-of-the-practice frameworks GERAM would therefore require substantial re-engineering. The IEEE 1471 [10] standard, now also adopted by the International Standardization Organization (ISO) and turned into the ISO/IEC 42010 standard [11] supports the enterprise architect with content requirements on architectural
descriptions. This standard composes a unification of best-practice solutions from a variety of architectural concepts. In contrast to GERAM, the core of IEEE 1471 is its focus on architectural support and guidance to supplement EA frameworks, especially in the domain of architectural descriptions. IEEE 1471 does not, however, go into any details concerning for instance EA processes. Based on this, the IEEE 1471 standard does not qualify as a meta framework.

From the generalization of frameworks we move on to comparative quality analysis by specified architectural purposes and goals. In research presented by Tang, Han and Chen [12] six frameworks are evaluated and compared according to their goals, inputs and outputs. The analysis provides an identification of differences between the frameworks based on the architectural process. However, by only focusing on the evaluation criteria and not on the framework as such, a meta view never materializes. Another work with similar scope is presented by Leist and Zellner [13]. At first they define main components important for an architectural method and from there evaluate a number of frameworks. Leist and Zellner focus strictly on methods for creating architectures. Many EA frameworks do discuss methods and processes for creating architectures, but this is rarely their main focus. Consequently, the framework of Leist and Zellner is inadequate for EA framework classification.

Urbaczewski et al. [14] review and compare some of the same frameworks as does the EAF2. Specifically, four frameworks are mapped to the Zachman framework’s views and abstractions. Although this gives an indication of how the different frameworks relate to the Zachman framework, it does not exhaustively describe their contents in a manner that facilitates a choice between the frameworks.

Abdullah et al. [15] have devised a framework for categorizing EA frameworks according to goals, inputs, outputs and miscellaneous other criteria. This framework is not merely descriptive; they devise a normative structure based on concepts external to the EA frameworks. Since the structure is not empirically validated it may or may not represent the actual requirements of the EA framework users.

Williams et al. [16] compare the relations of some frameworks to product life-cycles. The study is limited to enterprise architecture frameworks and life-cycle related issues only. Furthermore, the frameworks compared in the study (PERA, CIMOSA, GRAI-GIM and others) are strictly research-oriented and not currently in industry use.

Having concluded that the previously performed EA framework comparisons are all found wanting with respect to one or many requirements, we now proceed to detail how the EAF2 was developed.

III. THE PROCESS OF CREATING EAF2

The EAF2 was designed to reflect current practice within the EA discipline, as reflected by some of the most popularly used frameworks.

Another goal of the EAF2 is to span the space of different EA frameworks in use today. To achieve this, frameworks have been chosen that are in some sense "orthogonal" to each other, or at least cover different spaces. Thus, Zachman was chosen due to its famous taxonomy-based EA approach, TOGAF due to its architecture development process and FEA due to its prescriptive methodology for developing enterprise architectures. This spatial diversity is alluded to in Fig. 1, where the overall process of creating EAF2 is outlined. A more detailed description follows:

1. Selection of frameworks. The first step was to choose EA frameworks as a basis for EAF2. By choosing a number of frameworks frequently cited in academia or commonly used in industry a desirable width of application of the resulting meta framework was achieved. The frameworks included were TOGAF, DoDAF, IAF, MODAF, E2AF, FEA, the Zachman framework, and ArchiMate. According to [17], TOGAF, FEA, and Zachman, along with the Gartner methodology, are the most popularly adopted EA methodologies. Gartner was excluded from the EAF2 creation, as it was not fully accessible due to commercial restrictions. DoDAF, MODAF, E2AF and Archimate are intended to compensate for this.

2. Feature extraction. By analyzing the selected frameworks, their most important distinguishing features were identified and compared with each other. This resulted in a draft meta framework for enterprise architecture frameworks. The feature extraction method employed here featured four process steps, each with a set of accompanying rules, which are described below. The process steps were performed iteratively, addressing the frameworks one by one.

Step 1: Selection of a starting framework from the set of frameworks.

Step 2: Identification of entities to extract. The framework descriptions were scanned for figures,
tables, etc. to identify the key entities. In particular, entities that were either (i) given formal definitions, (ii) discussed in contrast with other frameworks, or (iii) recurred very frequently in the text, were chosen. **Step 3:** Grouping of the extracted entities. The extraction of each entity was accompanied by a classification; either into a class of similar entities, or as a unique one-of-a-kind entity. In the first iteration, every entity was a class of its own. After several iterations, when a larger number of frameworks had been subjected to analysis, fewer and fewer new classes emerged until no new classes emerged. **Step 4:** The process returned to step 1 with a new framework for feature extraction selected, until the set was exhausted.

3. Feature consolidation. The features extracted in the preceding step were refined and semantically identical concepts were merged, thus consolidating the EAF. The result of the preceding method step was a number of classes, many of which were very closely related, albeit not identical. To reduce the number of classes in EAF, an iterative process of consolidation was carried out:

**Step 1:** Selection of a class.
**Step 2:** Classes that resembled the initial class were compared to identify a superclass which would encompass all of the above classes. TOGAF’s Architecture Development Method, DoD’s Architecture Development Method, and the MODAF Architecting Process were, for instance, all consolidated into the EAF superclass Architecture Development Process as illustrated in Fig 2.
**Step 3:** The process returned to step 1, until the set was exhausted and all classes thus consolidated.

IV. THE EAF

The introduction of superclasses implies that the devised meta framework gets a hierarchical design where each level represents a set of these classes. EAF is divided into two main classes; Architecture Governance and (Enterprise) Modeling Concepts as illustrated in Fig. 3.

Architecture Governance [2], [3], [18], [19] describes the management aspects of EA, while Modeling Concepts contain definitions and formalisms for actual models. The latter class has also benefited from the GERAM description of Generic Enterprise Modeling Concepts [6].

Proceeding into greater detail, the Architecture Governance class is visualized in Fig 4. The most important parts of architecture governance are the two processes Architecture Development Process [2], [3], [18], [19] and Architecture Maintenance Process. The former one decomposes into Process Step [2], [3], [18], [19], [4], Architecture Techniques [19] and Continuous Improvement Process [2], [19], [4].

Process step is a straight-forward entity required to describe the step-wise development approach promoted by most frameworks. Architecture Techniques are used to design and implement an architecture by addressing information gathering, decision making and problem solving [19]. Continuous Improvement Processes stress that architecture process should continuously change to respond to the requirements of the enterprise. The final form of the EAF Architecture Development Process has also benefited from the GERAM description of Generic Enterprise Modeling Concepts [6].

Architecture Maintenance Process [4] is the process for maintaining and evolving EA models to keep them up-to-date with recent developments with the organization’s business and/or IT landscape.

Supporting these two main processes, architecture governance also contains a set of tools. Architecture Guidelines and Principles [2], [3], [19] are instrumental in keeping the construction of an organization’s architecture aligned with stakeholder’s requirements and can be seen as a support when making architectural decisions. Building blocks and Patterns [2], [19], [4] are the two main components. Our notion of Building blocks stems primarily from TOGAF and FEA. The building blocks provide units of functionality for fulfilling needs in an organization. Solution Alternatives group several building blocks into practical solutions that address organizational requirements. Patterns [2], [19] are more abstract descrip-

---

**Fig. 2.** Consolidation into a superclass.

**Fig. 3.** The topmost level of EAF.
tions of how to use building blocks, i.e. how they are put together into architectural solutions. The relationship between Building blocks, Patterns, and a few related concepts in the EAF² are illustrated in Table I.


The second main class in the meta framework is depicted in Fig. 5. The first important subset here is Model Taxonomy [2], [21], [1] which provides a way of structuring models used for enterprise modeling. The taxonomy defines which models an enterprise architecture can consist of and provides an overview of the models. The most widespread model taxonomy is the Zachman framework [1], which categorizes models in two dimensions, Abstraction levels [2], [19], [21], [1], [22] and Aspect areas [2], [19], [21], [4], [1], [22]. Another widespread taxonomy is the classification of architectural layers, viz. business, information, application and infrastructure.

The second of the modeling concepts is Reference models [2], [19], [18], [4], [6]. These are models capturing knowledge from previous modeling tasks, in a way similar to that of patterns. The main difference is that patterns are always normative, whereas reference models are sometimes purely descriptive, i.e. without endorsement showing how building blocks have been previously put together.

The final modeling concept is that of Metamodels [3], [18], [22], providing semantic rigor to EA frameworks. A metamodel formally defines the allowed contents of architectural models, often graphically. In EAF² the metamodel concept is made up from Entity Type [18], [22] and Relationship Type [18]. Entity Types represent architectural classes, e.g. information systems, processes, enterprises, and they are used to guide and delimit the contents and semantics of the EA models. Attribute Types [18], [22] are types of entity properties. Relationship Types [18], [22] define the legal entity connections. Viewpoints [2], [3], [19], [18], [21], [4], [1], [22] show excerpts of the metamodels according to the information needs of different stakeholders. The military architecture frameworks offer good examples of viewpoints, viz. the All Views, Operational View, System View, and Technical Standards View of DoDAF [3]
and the Strategic, Operational, System Acquisition, All Views, and Technical Standards viewpoints of MODAF [18].

Under these descriptions, the difference between Model taxonomies and Viewpoints is still somewhat vague. One way to clarify the difference is to consider the intended use of the different concepts. A taxonomy is primarily about classification. A good classification is exhaustive, i.e. it covers all cases, and disjoint, i.e. no case is classified into more than one category. A viewpoint, on the other hand, is a convenient presentation of information tailored for a specific stakeholder. While it is natural to expect that the set of viewpoints is exhaustive, there is no reason why the viewpoints should all be disjoint. Indeed, more often than not there exists some core information, relevant to many, or all, stakeholders. So, while taxonomies and viewpoints are similar, there are important differences as well.

### V. A Sample Classification of EA Frameworks

This section presents the classification of seven well known architecture frameworks using EAF$^2$ and the concepts presented in section 4. The seven frameworks are TOGAF [2], DoDAMF [3], MODAF [18], E2AF [21], FEA [4], Zachman [1], and Archimate [22]. These frameworks have been chosen based on wide-spread adoption and frequency of citations, together providing an example of how frameworks are classified using EAF$^2$.

The classification was done by matching the concepts in EAF$^2$ with related concepts from each framework. A three level scale was used, where a concept can be (i) presented and detailed, as done for instance in MODAF regarding the metamodel and all its sub concepts. The metamodel is a core concept of MODAF and is thus thoroughly detailed. A concept can also be (ii) mentioned and discussed, cf. TOGAF where patterns are introduced and the content of a pattern is defined. However the actual patterns "have not (as yet) been integrated into TOGAF" [2]. Finally a concept can be (iii) not mentioned at all in the framework to be classified, corresponding to a blank cell in Fig. 6.

### VI. Conclusions

This paper has demonstrated how the EAF$^2$ provides an easy and convenient way of contrasting different architectural frameworks with each other. By using the EAF$^2$, the choice of framework for any particular EA project or initiative is simplified and made more transparent. This is particularly important as different frameworks often relate to each other in a complementary fashion, a fact seldom mentioned by the proponents of the various frameworks.

It is clear that the concept "architecture framework", although abundantly used, is not very coherently defined. The table in the previous section demonstrates the usefulness of the EAF$^2$ classification for revealing the context dependency of the term. Some frameworks contain only a taxonomy, others focus on a metamodel, while some try to cover everything at the cost of less depth. While the EAF$^2$ does not resolve this ambiguity, it does provide a structured way to reason about it, and make rational decisions on matters of using enterprise architecture frameworks.

As for future work, an expansion of the present contribution to classify more frameworks by the EAF$^2$ standard would make the resulting catalog considerably more useful. Such an
expansion could also be used as a basis for a taxonomy of the terminology currently in use in different EA frameworks.

REFERENCES