A ZACHMAN CUBE

Vladan Jovanovic, Georgia Southern University, vladan@georgiasouthern.edu
Stevan Mrdalj, Eastern Michigan University, smrdalj@emich.edu
Adrian Gardiner, Georgia Southern University, agardine@georgiasouthern.edu

ABSTRACT

The Zachman Framework (ZF) is a matrix of distinct stakeholder perspectives and unique concerns or aspects of information system architecture. This work suggests how the ZF can be extended into a multi-dimensional Zachman's Cube. We illustrate its use with a set of UML usage styles, but one can choose other global system dimensions—for example, security.

Keywords: Zachman Framework, Zachman Cube, Enterprise Architecture Framework, UML

INTRODUCTION

Ever since its inception, the Zachman Framework (ZF) [15, 17] has provided organizational architects with an effective way of documenting enterprise-wide information systems architecture (ISA). An architecture framework is a standard for documenting what products must be delivered to whom and what concerns and subject matter they are addressing. The importance of an enterprise-wide architecture framework is widely recognized. Goethals [8] and Schekkerman [14] provide comprehensive overviews of enterprise architecture frameworks in the literature while Iyer and Gottlied [9] provide an informal comparison between specific architectures.

The problem of documenting 'multi-dimensionality' of architecture is also recognized in the software community and it has accumulated extensive experience in utilizing UML for documenting software systems architecture [2]. However, it is also struggling with similar challenges in mapping development technology to view-types and styles. For example, the IEEE P1016 [7] maps design languages, and UML in particular, to recommended viewpoints and concerns listing technology choices for each viewpoint. What is common to all of architecture frameworks, including ZD, is that they are not prescriptive in terms of modeling languages to be used nor the methodologies or processes.

As the ZF is an abstract two-dimensional framework independent of any methodology or tool, methodological and tool choices must be made before implementation. We assert, however, that it is difficult for organizational architects to visualize these choices, given that for each ZF cell, there may be several potentially unrelated (distinct) technologies. As a result, we also assert that a two-dimensional view fails to adequately represent the multidimensional design space facing most organizational architects. To address this conceptualization challenge, we introduce a Cube metaphor for the ZF. Extending the ZF by a new dimension will turn ZF into a 3D space and in the general case into an n-dimensional Cube. We view the ZF Cube as a significant extension to the original ZF and an important conceptual model to aid the documentation of enterprise-wide ISA.

Clearly, there may be numerous ways of encoding a technology dimension. In this paper we demonstrate the potential of the ZF Cube by limiting our focus to Unified Modeling Language (UML) and, in particular, its usage styles following the UML to ZF mapping analyzed by Frankel [6] and Mrdalj and Jovanovic [11]. UML as a language for documenting the modeling artifacts of systems is a technology that can address much of the ZF. However, even in organizations choosing to adopt the UML standard, architects have alternative usage styles to choose from. A multitude of choices within UML based technologies themselves can justify the extension of ZF into a Cube. We present these choices as discrete members of the technology dimension.

THE ZACHMAN FRAMEWORK AND UML

The aim of the ZF is to describe an architecture that represents information systems' artifacts within an abstract classification schema. Sowa and Zachman [15] described the ZF as taxonomy for relating things in the real world mapped to their computer representation forms: "*The ISA framework serves as a convenient classification scheme or "periodic table" for information entities."* Zachman's initial intention was for each matrix cell to be "normalized," with one fact in one place [18]. Given that Zachman considered the semantic distance between each cell to be significant, the ZF advocates a distinct conceptual modeling artifact for each cell. Zachman's goal, in this regard, is that if cells within the framework can

be considered primitive, they can be modeled or described independently. A weakness of the ZF, as taxonomy, is whether it can be considered complete. Another weakness is that significantly more effort is required by the architects to apply the ZF to real EA applications.

The Unified Modeling Language (www.uml.org/#UML2.0) is designed for visual representation of models and offers a field tested means to assess, build, and deploy information systems. Nowadays, the software development community has recognized UML as a standard specification language that has potentials beyond visual descriptions of software. West, Bittner and Glenn [17] state UML's applicability from business and data modeling to systems modeling. They add that UML is well suited for developing precise and complete visual descriptions of the elements of Enterprise Architecture (EA). The current version, UML 2, uses 13 diagram types: Class, Component, Communication, Interaction Overview, Composite Structure, Deployment, Object, Package, Sequence, State, Timing, and Use Case. It is important to note that diagram types are intended to be used pragmatically to fit the situation (problem) and approach taken. Elements from one diagram type occasionally may be used in another diagram. The UML descriptions can be used with a wide range of rigor and details befitting the intent such as a) sketches, b) detail engineering blueprints, and c) executable specifications, in the Model Driven Architecture (MDA) approach [10].

EXTENDING ZACHMAN FRAMEWORK

In order to illustrate a need for an extension of ZF, we addressed four publicly available empirical approaches of using UML:

- Rational Unified Process mapped into the ZF [3].
- The Popkin Process for Enterprise Architecture which is based upon the ZF and uses Popkin's tool System Architect (Popkin Software).
- ZF applied through the Select Perspective and UML using Select Component Architect [13].
- OMG's Model Driven Architecture mapping to the ZF [5].

Table 1 presents the combined mappings from the above four UML usage technologies onto cells of the traditional ZF. What is lost in the process of presenting distinct technology choices in an overlapping way? Obviously, we are losing the identity of any particular UML usage style, as ZF cells list all that apply as a union of the appropriate choices in individual technologies.

Unless we add a separate dimension, it is hard to represent the technologies all at once and still separate the individual technology choices per cell of the ZF. Such considerations were instrumental in our decision to extend the ZF into a Cube.

Table 1. All Technologies Mapped to the Traditional Zachman Framework

Zachman Framework	Data (What)	Function (How)	Network (Where)	People (Who)	Time (When)	Motivation (Why)
Scope Model	Class, Package, Use Case	Activity, Use Case, *Component*		Activity	Activity	
Enterprise Model	Class	Use Case, Activity, State, Sequence, Communication		Use Case, Activity	Use Case, Activity	
System Model	Class, Package, Component	Use Case, Activity, State, Sequence, Communication	Deployment, Component	Class, Use Case, Sequence	Sequence, Communication, State	Class
Technology Model	Class, Package, Component	Use Case, Activity, Class, Sequence, Component	Component, Deployment	Sequence, Class	Sequence, Communication, State	Package, Class, Sequence, Communication
Detail Representation			Component, Deployment		State, Sequence	Sequence

For each general type of information technology (IT), the extension proposed here will allow for discrete choices in a third dimension, as illustrated in Figure 1, and by analogy for any additional dimension. In situations typical for complex systems, the logic is recursive as per Sowa and Zachman [15]. This means that systems can be recursively described with evident types and subtypes of IT choices. The main point is that we are replacing separate choices on the individual cell level of the framework with an additional dimension—i.e., the whole system perspective. We consider a cube to be a true multidimensional representation that can be as easily extended beyond the third dimension. We illustrate the use of the ZF Cube as a series of related projections. There are five different **perspectives** of the system shown as rows in the ZF Cube. Each row can be seen as the stakeholder's view with the interrogative **aspects** and their projections into the technology dimension (Figure 2).

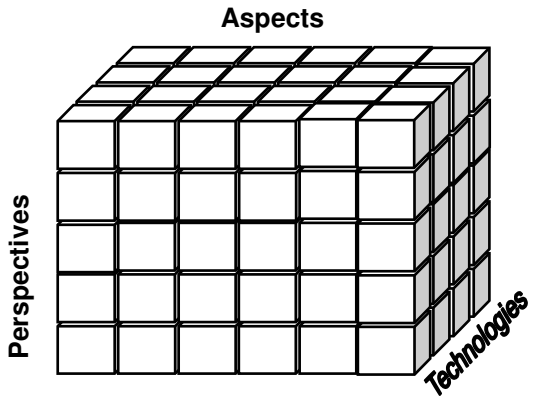

Figure 1. Zachman Cube

.

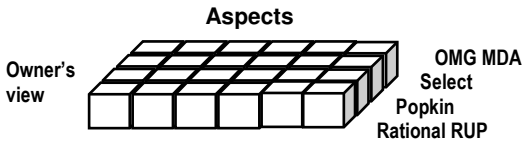

Figure 2. Technology Projections

Such projections are illustrated in Tables 2, 3 and 4, below. The content of those tables is adapted from the related work of Mrdalj and Jovanovic [11] that focuses on mapping the UML onto the ZF. The important distinction here is the recognition of the views as projections of the ZF onto the technology dimension, making the ZF into a ZF Cube. It is very important to separate out the generic information technology (IT) dimension from any particular ZF column's aspects, as each can be said to have its preferred 'technologies.' It could have been possible to present all six ZF perspectives, but for the purpose of illustration, we selected only views of general interest. Note also that the top row in each of the tables below serves only as a title header for the column projection on the subsequent four technologies.

Table 2 represents the Owner's View mapped onto the technology dimension. The owner is interested in the business deliverable and how it will be used. This provides a description of the organization within which the information system must function.

Table 2. Technology Projections of the Owner's view: Enterprise Model Artifacts

		Data	Function	Network	People	Time	Motivation
Technology Projections	**Rational RUP**	Class (Business classes)	Use Case (Business Use cases)		Use Case (Business Actors)	Use Case (Events)	
	Popkin		Activity				
	Select	Class (Business classes)	Activity (Process flow)		Activity (Business Actors)	Activity (Events)	
	OMG MDA	Class	Activity, State, Sequence, Communication				

Table 3 represents the Designer's View mapped onto the technology dimension. This view outlines how the system will satisfy the organization's information needs. The representation is free from solution-specific aspects or production-specific constraints. It was somewhat surprising to us to see that the dialog design interaction was not captured using state charts. This is not meant as a criticism but rather to indicate the incompleteness and openness of proposed mappings to an individual organization's practices, processes and methodologies.

Table 3. Technology Projections of a Designer's View: System Model Artifacts

		Data	Function	Network	People	Time	Motivation
Technology Projections	**Rational RUP**	Class (Persistent classes)	Use Case (Realization)	Deployment	Class (Boundary classes)	Sequence, Communication	Class (Multiplicities)
	Popkin	Class	Use Case	Component	Use Case		
	Select	Class	Use Case	Deployment, Component	Sequence, Class	State	
	OMG MDA	Class, Package, Component	Activity, State, Sequence, Communication	Deployment			

Table 4 represents the Builder's view mapped onto the technology dimension. This view provides a representation of how the system will be implemented. It makes specific solutions and technologies apparent and addresses production constraints.

Table 4. Technology Projection of Builder's View: Technology Model Artifacts

		Data	Function	Network	People	Time	Motivation
Technology Projections	**Rational RUP**		Component				
	Popkin	Class	Use Case, Activity	Deployment		Sequence, Communication, State	
	Select		Use Case, Class, Sequence, Component	Component, Deployment	Sequence, Class	Sequence, Communication	Package, Class, Sequence, Communication
	OMG MDA	Class, Package, Component	Activity, State, Sequence, Communication	Deployment			

CONCEPTUAL RATIONALIZATION AND FORMALIZATION OF THE ZACHMAN CUBE

In this section we discuss the formal and conceptual needs for ZF extensions. For this purpose we developed a class model for mapping the UML to the ZF illustrated in Figure 3. Consider that the 'corresponds' association from Figure 1 has unlimited multiplicities on both sides. The implication is that a number of UML artifacts can be useful in a number of ZF 'cells.' After this observation one should make room for additional dimensions on top of stockholder's perspectives and system aspects of the traditional ZF.

The separation of concerns, as a principle, provides for a clear framework for conceptualization and selection of technologies, but the originally intended restriction of the ZF (one unique and exclusive deliverable for each cell) is unattainable as acknowledged in alternative architectural frameworks and standards [7, 12].

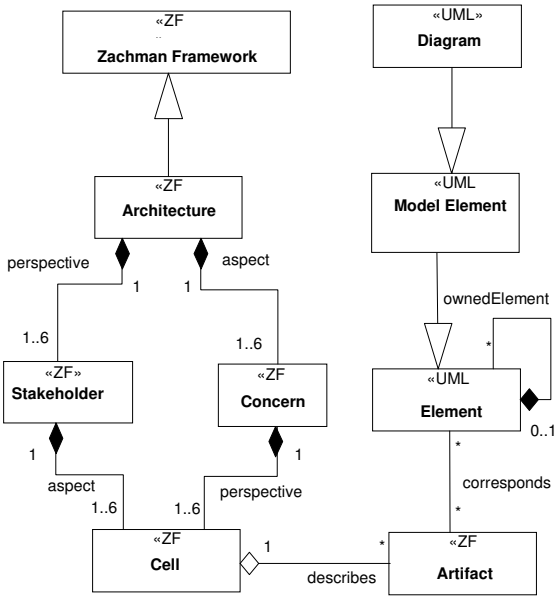

Figure 3. UML Mapping to ZF

Subsequently, it is not supposed to preclude people of thinking in terms of 'unexplored' relationships leading to more integrated systems. The obvious examples are the potential uses of the Class diagram which expresses the concept of an object in terms of the interrogatives it addresses and the perspective it serves. For example, since an object is the encapsulation of data, process, and rules, then a deliverable comprising such objects fits into Row 3, and Columns 1, 2, and 6.

A similar alternative framework was developed as early as 1992 by Bruce [1] with object-oriented specific aspects, but we decided to extend the current ZF with an information technology dimension instead of revising the established aspect of the ZF with each new technology paradigm. For example, consider the obvious consequence of Aspect Orientation (AO), where the point is separating aspects representing cross-cutting concerns [4]. Once such aspects are recognized, it is almost impossible to keep all concerns fully independent. All approaches mentioned in this paper struggled with the multidimensionality of architectural representations. An approach used in IEEE P1016 [7] to define meta entities, relationships and constraints in each viewpoint corresponds to subject matter entities in each of the cells in the ZF. Such a generalized approach explicitly opens up architectural frameworks for technological choices of interest to the systems design community. In the case of the ZF, this opens up possibilities that are obviously related on how to fit the 'whole technology set' into a framework. Consequently, the role of standardization in the entire scope of the multidimensional ZF, i.e., the ZF Cube, becomes paramount.

CONCLUSION

Other researchers also recognized a need to expand the ZF. Iyer and Gottlieb [9] used four domains (process, information, infrastructure and organization) to effectively drill down into the ZF cells. Our approach extends the ZF into a Cube, adding new dimensions over all the cells. We decided to use the four UML based design approaches as a technology dimension. This illustration was selected due to the UML's importance in Systems Design. While further 'dimensions' are also possible, we argue that our illustrations are sufficient for the aim of showing how to construct and use the ZF Cube

We also learned that most technologies will not cover the entire ZF scope—not making technology choices any easier, as such partially applicable technologies have to be patched and integrated. What is important here is that we do not have completely satisfying approaches even if we focus on the most comprehensive language like UML. Therefore, organizations should be well advised not to relinquish the responsibility too easily to 'outsourcing solution providers,' particularly given the immature and open nature of development technologies. The fact is that with tools like the ZF Cube, we can see the scope of the problem, but organizations remain responsible for the selection and improvement of know-how for their quality design work.

REFERENCES

1. Bruce, T. (1992). *Designing Quality Databases with IDEF1X Information Models*, New York: Dorset House Publishing.
2. Clements, P. et al. (2003). Documenting Software Architectures: Views and Beyond. Upper Saddle River, NJ: Addison Wesley.
3. De Villiers, D.J. (2001). Using the Zachman Framework to Assess the Rational Unified Process. *The Rational Edge*, March 2001. Available: 128.ibm.com/developerworks/rational/library/372.html.

4. Filman, R., *et al* (2005). *Aspect-Oriented Software Development*, Upper Saddle River, NJ: Addison Wesley.
5. Frankel D.S., *et al* (2003). The Zachman framework and the OMG's model driven architecture, *Business Process Trends*, September 02, Available: www.bptrends.com.
6. Frankel D.S., et al. (2003b). *Model Driven Architecture—Applying MDA to Enterprise Computing*, Hoboken, NJ: Wiley Publishing.
7. IEEE (2006). Standard P1016 Software Design Description, working draft 5.0, IEEE.
8. Goethals, F. (2003). An Overview of Enterprise Architecture Framework Deliverables. May 2003. Avalable: www.econ.kuleuven.ac.be/ leerstoel/sap/FramesPage.htm.
9. Iyer, B. & Gottlieb, R. (2004). The four-domain architecture: An approach to support enterprise architecture design, *IBM Systems Journal, 43*(3), 587-597.
10. Mellor, S & Balcer M. (2002). *Executable UML: A Foundation for Model-Driven Architecture*, Upper Saddle River, NJ: Addison Wesley.
11. Mrdalj, S. & Jovanovic, V. (2005). Mapping the UML to the Zachman Framework. *Proceedings of the Eleventh Americas Conference on Information Systems*, Omaha, NE, August 11-14.
12. Putman, J. (2001). *Architecting with RM/ODP*, Upper Saddle River, NJ: Prentice Hall.
13. Select Products, *Zachman Framework to Select Products Mapping*, Available: www.selectbs.com/products/solutions/zachman_ framework.htm.
14. Schekkerman, J. (2003). *How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework.* (2nd edition), Oxford, UK: Trafford Publishing.
15. Sowa, J.F. & Zachman, J.A. (1992). Extending and formalizing the framework for information systems architecture, *IBM Systems Business Journal*, *31*(3), 590-616.
16. Spewek, S.H. (1993). *Enterprise Architecture Planning*, Boston, MA:. QED Publishing Group.
17. West, D., Bittner, K. & Eddie Glenn E. (Nov. 2002). Ingredients for Building Effective Enterprise Architectures, *Rational Edge*. Available online at http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/nov02/EnterpriseArchitectures_TheRationalEdge_Nov2002.pdf
18. Zachman, J. (1987). A framework for information systems architecture. *IBM Systems Journal, 26*(3), 276-292.