Apply ADD method to the Development Process Life-Cycle Phases within DoDAF

Bo Ou[1, a], Daguo Qin[2, b]

[1,2]Equipment of Academy, Beijing 101416, China

[a]oubo@china.com, [b]qdg2000@m165.com

Keywords: Attribute-Driven Design (ADD), Architectural Description, Life-Cycle, DoDAF.

Abstract. This paper presents an approach to work with the process life-cycle of the U.S. Department of Defense Architecture Framework (DoDAF), and extend existing architecture description process and artifacts within DoDAF that match the scope of the ADD life-cycle phases to enhanced system life-cycle development process that includes process-iteration based development and determination in an integral manner.

1. Introduction

The goal for the U.S. Forces, like other militaries around the world, are putting renewed efforts into C^4ISR to find a way which can collects, analyzes, disseminates, and shares information gathered to provide commanders with trusted and relevant information for decision making and enhancing their own command and control. But C^4ISR is a large-scale systems, many design activities occur in parallel with each parallel activity developing a portion of the architecture. These separate design pieces must be merged (and possibly changed) to create a consistent and complete architecture that meets all the stakeholder requirements, Some architectures are developed under a set of guides called Architectural Framework, which are a set of practices and knowledge that enable organizations to build their own architecture, the Department of Defense Architecture Framework (DoDAF) was one of them.

Following DoDAF thorough iterations produces models, reports and analysis (artifacts) that form architectural building blocks. Such architectural elements provide support to the organization's strategic business and decision-making processes, based on ICT resources and structures. It is necessary to ensure that the artifacts that describe different aspects of the organization from different perspective are correct and consistent with reality. For this purpose have been developed different methods for Architectural Development Process (ADP), one of them is Attribute-Driven Design (ADD) which can documents an architecture in a number of model views.

2. Concept and definition

2.1 DoDAF

The DoDAF is mandated for expressing high level system and operational requirements and architectures that cross organizational and national boundaries[1]. The principal objective of DoDAF is to ensure that architecture description can be compared and related across organizational boundaries, by defining a particular set of architectural elements and relationships used for describing architectures.

Figure 2.1-1 depicts the high-level, 6-step architecture development process.

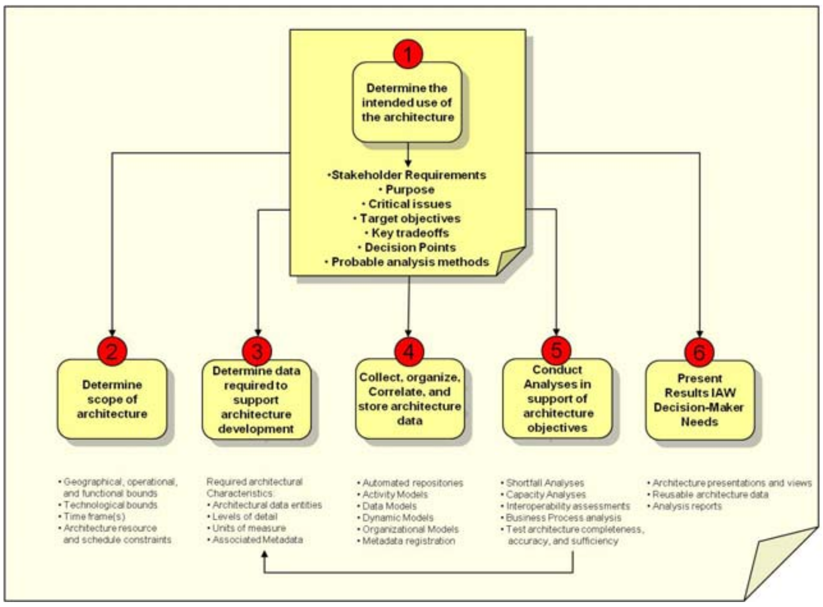

1
Determine the intended use of the architecture
•Stakeholder Requirements
• Purpose
• Critical issues
• Target objectives
• Key tradeoffs
• Decision Points
• Probable analysis methods
2
Determine scope of architecture
3
Determine data required to support architecture development
4
Collect, organize, Correlate, and store architecture data
5
Conduct Analyses in support of architecture objectives
6
Present Results IAW Decision-Maker Needs
• Geographical, operational, and functional bounds
• Technological bounds
• Time frame(s)
• Architecture resource and schedule constraints
Required architectural Characteristics:
• Architectural data entities
• Levels of detail
• Units of measure
• Associated Metadata
• Automated repositories
• Activity Models
• Data Models
• Dynamic Models
• Organizational Models
• Metadata registration
• Shortfall Analyses
• Capacity Analyses
• Interoperability assessments
• Business Process analysis
• Test architecture completeness, accuracy, and sufficiency
• Architecture presentations and views
• Reusable architecture data
• Analysis reports

Figure 2.1-1

2.2 Attribute-Driven Design

ADD is an architecture design method "driven" by quality attribute concerns – Version 2.0 released November 2006. The method promotes an iterative approach to design. It provides a detailed set of steps for architecture design – enables design to be performed in a systematic, repeatable way – leading to predictable outcomes (in Figure 2.2-1). Inputs to ADD are functional requirements, design constraints, and quality attribute requirements that system stakeholders have prioritized according to business and mission goals.[3]

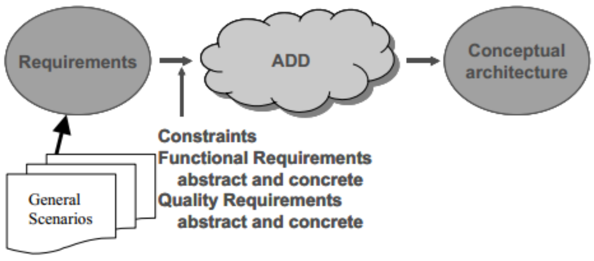


Figure 2.2-1

3. Current state of DoDAF and problem identification

DoDAF's modeling framework (MF) does not contain explicit deliverables for Identification, Implementation and Operation life-cycle phases. Once the Architectural Description Purpose and Scope are identified, within the 6-Step Architecture Development Process, in Step 3 the architect determines the data needed to support the Architectural Description development. In each step, the Meta-model Groups referred to by the step is that data in the Meta-model Groups [2]. Figure 3-1 depicts the sub steps that the architect needs to perform within the 6-Step Architecture Development Process. Some of these sub steps are performed in concert with the decision-maker, but the architect has more steps than the decision-maker.

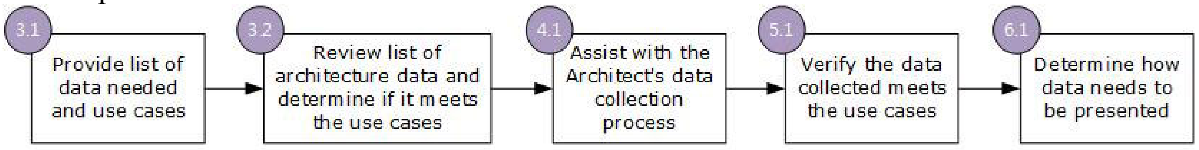

3.1
Provide list of data needed and use cases
3.2
Review list of architecture data and determine if it meets the use cases
4.1
Assist with the Architect's data collection process
5.1
Verify the data collected meets the use cases
6.1
Determine how data needs to be presented

Figure 3-1

While working with the DoDAF AF described in 6-step architecture development process, we made the following observations:

1. The architect doing the development must be familiar with the C^4ISR concerns and alternative patterns used in the project. This architect was also familiar with the ways of reasoning about selecting between alternatives and the timing model needed to evaluate the effectiveness of the choices.

2. The documentation structure will not be dependent on the development method but rather on the most effective way of capturing the views developed.

3. The developer chose to develop the architecture in a single iteration, which resulted in too many alternative patterns to represent comfortably as a matrix [3]. The pros and cons of each pattern were also not detailed explicitly, but were embedded within the rationale for making a selection within each pattern alternative.

4. DoDAF weakly implies that the development of an architecture is done sequentially—at each iteration, an element is chosen for design elaboration, all architectural drivers are known before starting the design of the element, and this iteration's results are then used in the next iteration. In development of large-scale architectures like C^4ISR, this in unlikely to happen. Different architects (or architecture teams) will be assigned to different elements of the architecture and will work in parallel; that situation will require cooperation between the architects working on different elements and an eventual merging of the resulting designs.

4. Propose a solution

DoDAF AF includes a number of activities that logically belong to different parts of the ADD approach. Because DoDAF is designed for DoD managers at all levels to make key decisions more effectively through organized information sharing across the Department, Joint Capability Areas (JCAs), Mission, Component, and Program boundaries [2]. Many of these activities can be broken down into sub activities, and ADD can follows a recursive design process to decompose a system or system element by applying architectural tactics and patterns that satisfy its driving requirements [5]. As illustrated in Figure 4-1, ADD essentially follows a "Plan, Do, and Check" cycle:

Plan: Quality attributes and design constraints are considered to select which types of elements will be used in the architecture.

Do: Elements are instantiated to satisfy quality attribute requirements as well as functional requirements.

Check: The resulting design is analyzed to determine if the requirements are met. This process is repeated until all architecturally significant requirements are met.

This process is repeated until all architecturally significant requirements are met.

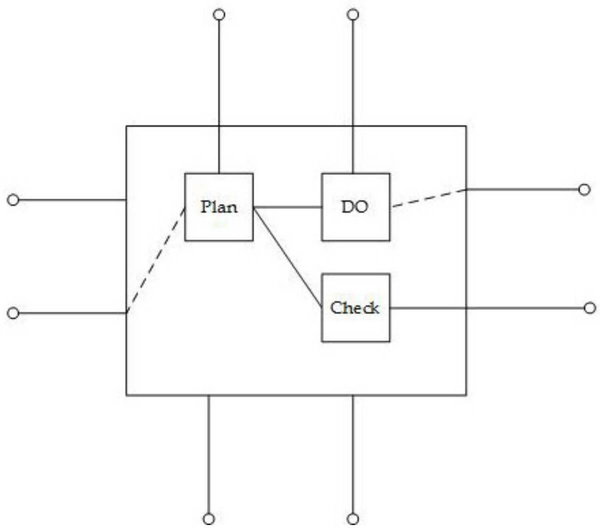


Figure 4-1

5. Details of development of the DoDAF based on ADD's life-cycle concept

5.1 Preparation of basic data requirement and data model

5.1.1 Quality Attribute Scenarios

We can export the quality attribute from the Joint Capability Area (JCA) which is part of DoD, Enterprise Architecture (EA). And group the priorities based on both their importance to stakeholders and their architecture impact; that is, now we use simple high (H), medium (M), and low (L) rankings to describe the priority of requirements, we have 9 groups: (H, H) (H, M) (H, L) (M, H) (M, M) (M, L) (L, H) (L, M) (L, L) [5].

Table 5.1.1-1

ID	Quality Attribute	Priority
QA-1	Force Application	H, H
QA-2	Building Partnerships	H, M
QA-3	Command & Control	H, H
QA-4	Net Centric	H, H
QA-5	Battle space Awareness	M, M
QA-6	Protection	M, L
QA-7	Logistics	M, M
QA-8	Force Support	H, L
QA-9	Corporate Management Support	M, L

5.1.2 General Scenario

We want to articulate what it means to achieve an attribute by identifying the yardsticks by which it is measured or observed. To do this, we introduce the concept of a "general scenario" Each general scenario consists of the stimuli that requires the architecture to respond, the source of the stimuli, the context within which the stimuli occurs, the type of system elements involved in the response, possible responses, and the measures used to characterize the architecture's response

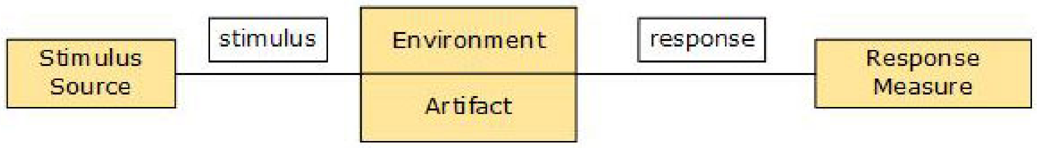


Figure 5.1.2-1

We describe a scenario with six factors for example as follows:

Table 5.1.1-2

Quality Attribute Scenario Detail	
Element	**Statement & View**
Stimulus	SvcV-4, SV-4
Stimulus source	SvcV-5, SV-5
Environment	SvcV-1
Artifact	OV-4
Response	SvcV-10b, SV-10b
Response measure	SvcV-7, SV-7

5.1.3 Constraints

We export the constraints from the scope of architecture as follows:

Table 5.1.2-1

ID	Constraints
CON-1	Geographical operational and functional bounds.
CON-2	Technological bounds.
CON-3	Time frame(s)
CON-4	Architecture resource and schedule constraints.

5.2 Migration of the development process life-cycle from DoDAF to ADD as Figure 5.2-1 described.

We begin by presenting the steps performed when designing a conceptual architecture using the ADD method. We then discuss the steps in more detail.

1. We have collected sufficient requirement information for the input of ADD. These information comes from stakeholder's requirement, purpose, critical issues, target objectives, key tradeoffs, decision points or probable analysis method.

2. We starts the decomposition typically with the top most design element if there is a Greenfield system. Otherwise the design element to start with is usually the whole system. All required inputs for this design element should be available (constraints, functional requirements, quality requirements) [6] from the Table 5.1.1-1, 2, 5.2.1-1.

3. Requirements in the (H, H) group are highly important to the stakeholders and are expected to have a high impact on the structure of the architecture, and so forth. From these pairs, you should choose several (five or six) high-priority requirements as the focus for subsequent steps in the design process. Then as we focus on data processing, we must keep data model to be consistent throughout all the life-cycle of development.

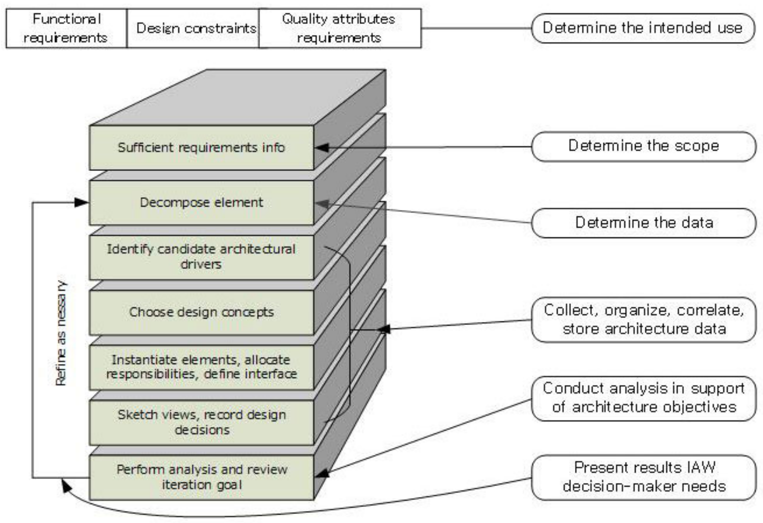
4. We should choose the major types of elements that will appear in the architecture and the types of relationships among them and determines the appropriate collection methods for the "Fit-for-Purpose" needs.

5. Using the identified Meta-model Groups in the DM2, the architect determines the method to collect the data and instantiate the various types of elements chosen in the previous step. Instantiated elements are assigned responsibilities according to their types [8].

6. Using the identified Meta-model Groups in the DM2, the architect determines the usage of the data and record the findings in the interface documentation for each element. The interface is not simply a list of operation signatures. Interfaces describe the PROVIDES and REQUIRES assumptions that elements make about one another.

7. Verify that the element decomposition thus far meets functional requirements, quality attribute requirements, and design constraints and prepare child elements for further decomposition. Such analysis should be the joint responsibility of the stakeholders and the architect to ensure it answers the stakeholders' questions.

8. As we have finished 7 steps in life-cycle phases, we have a decomposition of the parent element into child elements. Each child element is a collection of responsibilities, each having an interface description, functional requirements, quality attribute requirements, and design constraints [9]. If we have more components do decompose we return to step 2, otherwise we will present result which decision-maker needs.



Functional requirements
Design constraints
Quality attributes requirements
Determine the intended use
Sufficient requirements info
Determine the scope
Decompose element
Determine the data
Identify candidate architectural drivers
Choose design concepts
Collect, organize, correlate, store architecture data
Instantiate elements, allocate responsibilities, define interface
Refine as nessary
Sketch views, record design decisions
Conduct analysis in support of architecture objectives
Perform analysis and review iteration goal
Present results IAW decision-maker needs

Figure 5.2-1

6.Conclusion

Under a DoD mandate, DoDAF specifications will become the basis for all information system design in the near future. Although the current DoDAF specification provides an extensive methodology for system architectural development, it is deficient in several related dimensions – absence of ADP life-cycle management, especially for process-iteration throughout the development process [10], and lack of associated determination principle support. To overcome these deficiencies, we described an approach to support specification of DoDAF architectures within a life-cycle management based on ADD. The result is an enhanced system life-cycle development process that includes process-iteration based development and determination in an integral manner.

References

[1]. DoD Architecture Framework Working Group. DoD Architecture Framework Version 2.0 Volume I : Architect‘s Guide [R]. The United States: Department of Defense , 2009 May 18th.

[2]. DoD Architecture Framework Working Group. DoD Architecture Framework Version 2.0 Volume II: Architect‘s Guide [R]. The United States: Department of Defense , 2009 May 18th.

[3]. Wojcik, R.; Bachmann, F.; Bass, L.; Clements, P.; Merson, P.; Nord, R.; & Wood, B. *AttributeDriven Design (ADD), Version 2.0* (CMU/SEI-2006-TR-023). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006. p.7

[4]. Hofmeister, C., Nord, R., Soni, D. Applied Software Architecture, Addison Wesley, 2000.

[5]. Bass, L.; Clements, P.; & Kazman, R. *Software Architecture in Practice*. Reading, MA: Addison-Wesley, 2003.

[6]. Clements, P.; et al. *Documenting Software Architectures Views and Beyond*. Reading, MA: Addison-Wesley, 2003.

[7]. DOD Instruction 5000.2 “Operation of the Defense Acquisition System,” 12 May 2003.

[8]. Avgeriou, P., Zdun, U.: Architectural Patterns Revisited - a Pattern Language. In: 10th European Conference on Pattern Languages of Programs, Irsee, Germany (July 2005)

[9]. Burnstein, I.: Practical Software Testing. Springer, Heidelberg (2003)

[10]. Hofmeister, C., Nord, R., Soni, D.: Applied Software Architecture, pp. 7–8. AddisonWesley, Reading, MA (2000)