

Using ArchiMate with TOGAF

Part 2: Aligning the entities they use to document architecture

Graham Berrisford and Marc Lankhorst

Introduction

We started talking with the idea we should minimise ambiguity in our students' minds about when and where they should use the ArchiMate box symbols to represent those architecture entities that are named in an architecture method (be it TOGAF or any other method).

This paper is the fourth in a series, edited from our conversation over several months, in which we explore the possibilities of using the ArchiMate language with an architecture method, and in particular with TOGAF. The series consists of the following papers:

- [1]: ArchiMate is not just a set of box and line symbols, it has a meta model and philosophy that may or may not match those of any given architecture method. The first paper provides a kind of method maturity model, which you can use to assess the suitability of your architecture method for use with the ArchiMate language.
- [2]: The second paper applies the model in the first paper to an example architecture method. It shows that TOGAF supports some ArchiMate distinctions more clearly than others. And that the two approaches feature different kinds of "realisation" transformation.
- [3]: The third paper shows that ArchiMate draws a structure-behaviour dividing line in a different place from the ISEB reference model and from the TOGAF meta model. It also explores how people use the term Function loosely and generically where the terms Service, Process, Interface and Component could be used more precisely.
- This paper: Using ArchiMate box shapes to draw diagrams is trivial; it does not mean you are using your chosen architecture method in accord with ArchiMate's meta model and philosophy. This fourth paper provides the kind of careful entity-by-entity analysis that is needed if you want to use ArchiMate to support your chosen architecture method. Again, TOGAF is the chosen example.

In this paper, we investigate the mapping between ArchiMate's concepts and TOGAF's implicit and explicit meta models along the following lines. First, we ask whether TOGAF's meta model aligns with ArchiMate's. We investigate the generic, meta meta level of both and go deeper into the business architecture, data architecture, applications architecture and technology architecture as prescribed by TOGAF, to see whether we can express the required TOGAF notions in ArchiMate. We conclude with recommendations to the designers of both ArchiMate and TOGAF.

Classification of architectural entities

This section uses a tabular framework to introduce the ArchiMate and TOGAF entities. The tabular framework is only a presentation device. ArchiMate does not subdivide the active structure into logical and physical; that subdivision is an artificial device we have introduced to make comparison with TOGAF easier.

The first table shows the ArchiMate's architectural entities we are going to discuss.

Passive structure	Behaviour	Logical active structure	Physical active structure
Business			
Product			
Contract	Business Service		Business Interface
Business Object	Business Process / Function	Role	Actor
Applications			
	Application Service		Application Interface
Data Object	Application Function		Application Component
Infrastructure			
	Infrastructure Service		Infrastructure Interface
Artifact		Node	Device Sys Software Communication Path Network

The second table shows the TOGAF architectural entities we are going to discuss. Notice the emphasis on defining logical components.

Passive structure	Behaviour	Logical active structure	Physical active structure
Precursors			
Driver			
Goal, Objective			
Measure	Service Quality		
Business			
Contract	Service	Capability	
Product	Process	Function	Organization Unit
Location	Control	Role	Actor
Data			
Data Entity	Event	Logical Data Component	Physical Data Component
Applications			
	Information system service	Logical Application Component	Physical Application Component
Technology			
	Platform Service	Logical Technology Component	Physical Technology Component

Does TOGAF's meta model align with ArchiMate's?

Marc: Does the TOGAF meta model align with ArchiMate's?

Graham: Hmm... You cannot fully understand TOGAF by studying its explicit meta model. As a new arrival in TOGAF 9 (derived from the meta model in SAP's EAF) it does not yet fully match the implicit meta model in the TOGAF text.

Marc: The Open Group Architecture Forum has to address its own challenges. Some politics are reflected in TOGAF's internal inconsistencies. Convergence of explicit and implicit meta models is a challenge the Architecture Forum needs to resolve, ideally before convergence with ArchiMate.

However, the explicit TOGAF meta model doesn't look too far removed from ArchiMate's meta model. The table below gives us a starting point.

First-cut entity mapping table (needing further research)	
ArchiMate Entities	TOGAF Entities
Generic (meta meta) level	
Service Interface	Service Contract
Behaviour element	Function Process
Structure element	Function
Business architecture	
Actor	Actor Organisation unit
Role	Actor Role
Business Function	Function Capability
Product Business Object	Product
Contract	Contract
Business Service Business Interface	Business Service
Data architecture	
Data Object	Data Entity
	Logical Data Component
	Physical Data Component
Application architecture	
Application Component	Application Component
	Logical Application Component
	Physical Application Component
Application Service Application Interface	Information System Service
Infrastructure or Technology architecture	
Node Device System Software Artifact	Technology Component
	Logical Technology Component
	Physical Technology Component
Infrastructure Service Infrastructure Interface	Platform Service

Graham: The table is still superficial. There are two challenges here.

- First, the table compares only meta models. It relies on the new TOGAF meta model definitions. But to the TOGAF practitioner, it is not always clear how ADM artifacts and outputs are constructed using the entities in this new meta model.
- Second, some of the mappings are questionable. Mappings that look obvious at first glance may disintegrate on closer inspection. We need to do a thorough entity-by-entity comparison, comparing the entities' definitions, attributes and relationships. And at the same time consider effect of recursive decomposition on the entity definitions.

I have posted some slides that offer a more visual comparison of the meta models at [7]. We

should now go on to do a more detailed analysis, comparing entities in the two meta models, in the next sections.

Preliminary remarks

Entity definitions and attributes

Graham: TOGAF offers definitions for all 35 of its entities. TOGAF does list attributes for entities, but most of them are generic attributes and not very descriptive of a particular entity. Two of the entities, Contract and Physical Application, do have really substantial attribute lists it is worth looking up.

Marc: ArchiMate offers definitions for all of its 30 entities. ArchiMate does not list attributes for these entities, but includes a mechanism for assigning your own attributes to them.

Entity decomposition

Graham: Some (curiously not all) TOGAF entities can be hierarchically composed and decomposed. The obvious examples are function and process. Does ArchiMate allow entities to be composed and decomposed to any number of levels?

Marc: Yes. Any ArchiMate entity can be subdivided into sub-entities of the same type, ad infinitum.

Graham: Decomposition of entity types often undermines relationships that are implied in text definitions of the entities. E.g. consider that:

- ArchiMate appears to map one function to one role.
- TOGAF appears to map one business service to one application component.

These mappings cannot hold true between entities at different levels of recursive decomposition. And they are true at one level only if our method makes us force fit one concept to the other.

Architecture precursors

Marc: ArchiMate lacks modelling concepts for goals, objectives and requirements. We initially decided to leave these out of the modelling language, because we observed that people tended to describe these in natural language and not in models. But I expect that concepts for these will be added to a next version of the language, and preliminary ideas on this are already being formed. See also the remark on Reason below.

Graham: An architecture language without business goals is not really an enterprise architecture language. I think it is fair to say TOGAF captures architecture requirements at four levels of elaboration:

- business/IT goals
- business/IT objectives (with KPIs)
- architecture requirements (with measures)
- logical architecture building blocks (with non-functional qualities)

ArchiMate-only concepts

Graham: I notice ArchiMate includes the entities Meaning, Value, Interaction and Collaboration, which don't resemble TOGAF entities.

Marc: Indeed. The first two, Meaning and Value, are used to capture semantics. Meaning refers to the knowledge or expertise contained in (the representation of) a business object. You might call that "business semantics". It can be used as a 'hook' for inclusion of more semantics-based modelling and reasoning, which is certainly a consideration for future extensions of the language. Value is that which makes some party appreciate a service or

product, possibly in relation to providing it, but more typically to acquiring it. It helps in linking ArchiMate to techniques for modelling value networks and business models.

Next to Meaning and Value, there is a third 'semantic' concept hidden in the meta structure, which is not available to language users: Reason. This is intended as a generic, meta level basis for requirements modelling concepts, which can be defined as specialisations of Reason. As said, at the time we found that people tended to use natural language and not models for describing requirements, but in hindsight, we should perhaps have made these concepts available from the start.

Interaction and Collaboration are a pair of concepts to model cooperative structures. A Collaboration groups Roles that work together, and an Interaction is used to model what they do together. With these, you can model cooperation in a more symmetrical fashion than with services (which are of course inherently bound to a provider-consumer view of the world). This is for example useful in high-level models of the context of an organisation.

Generic schema (meta meta model)

Graham: TOGAF does not have a meta meta model, but we can draw some loose correspondences between ArchiMate's higher-level meta entities and TOGAF's entities.

Service and Interface

ArchiMate's Service	TOGAF's Service
Service A unit of essential functionality that a system exposes to its environment, which provides a certain value (monetary or otherwise), which thus provides the motivation for the service's existence.	An element of behaviour that provides specific functionality in response to requests from actors or other services. A service delivers or supports business capabilities, has an explicitly defined interface, and is explicitly governed. Services are defined for business, information systems, and platforms.
	TOGAF's Contract
	An agreement between a service consumer and a service provider that establishes functional and non-functional parameters for interaction.
ArchiMate's Interface	
Abstract location where a Service can be accessed by its users.	

Marc: It looks like a TOGAF Business Service somehow combines ArchiMate's Business Service and Business Interface. Perhaps a TOGAF Contract is more related to Service than Interface.

Graham: The separations of and relationships between Service, Interface and Contract are unclear. There is potential confusion between levels of granularity, and between process and system. A TOGAF Service appears to be a fine-grained elementary service (the result of a single process invocation). But a TOGAF Contract's attributes could describe an elementary service and/or a service level agreement for whole system or subsystem.

Marc: You could certainly put all of the Contract attributes in a SLA for whole system, but to check conformance to that SLA, you would also need to record some of these same attributes for each single invocable elementary service.

An ArchiMate service may be offered through various interfaces, and vice versa, the same interface may offer different services. An ArchiMate Interface is basically an "access point" for a service. At the business layer, this includes e.g. the telephone, post, email, web, etc.. All can be used for accessing services (and/or products) from the enterprise. At the application level, we might have on the one hand user interfaces of software systems, and on the other hand machine-to-machine interfaces (e.g. web service stuff). At the infrastructure level, you'll have lower-level software or hardware interfaces.

Graham: It seems to me both ArchiMate and TOGAF ought to separate as many as six concepts in this area:

1. Data flow: A transient data store passed from a "sender" component to one or more "receiver" components, which contains some data. E.g. a serial file, a message, a text document.
2. Data flow content (cf. an ArchiMate object): A data structure definable separately from the data flow that contains it. Definable as a regular expression, perhaps using an XML schema. E.g. a data entity or aggregate of data entities.
3. Interface (in the ISEB sense): An aggregate of services assignable to one or more "server" components for use by one more "client" components.
4. Channel (cf. an ArchiMate Communication Path): Used to transmit a data flow (1) or make a client-server connection (3). E.g. telephone, HCI, internet, private network.
5. Protocol: One or more layers of protocols needed to send a data flow (1) or invoke services (3) via a channel (4).
6. Location: an end point for a Data Flow or Channel.

Behaviour and structure

ArchiMate's Behaviour Element	TOGAF's Function
Unit of internal behaviour that can be composed of other Behaviour Elements and realise Services.	Delivers business capabilities closely aligned to an organization, but not necessarily explicitly governed by the organization. Also referred to as "business function".
	TOGAF's Business Process
	A process represents flow of control between or within functions and/or services (depends on the granularity of definition).
ArchiMate's Structure Element	TOGAF's Function
Basic unit of active structure that can perform the Behaviour Elements assigned to it and provide interfaces to the Services these Behaviour Elements realise.	(As above)
ArchiMate's Object	
Basic element on which behaviour is performed.	

Marc: ArchiMate uses a meta meta entity "Behaviour Element" to abstract from the various meanings people give to "Function". Similarly "Structure Element" abstracts from meanings associated with "Component".

Graham: It would of course be a circular argument to justify a structure-behaviour distinction by reference to meta meta entities we have created to match the distinction we draw. If our aim is reduce practitioners' confusion between the different kinds of function that they are thinking of and working with, then talking about even more abstract super types doesn't feel like a promising path! The fundamental system modelling distinction I miss in ArchiMate is the process/component (or procedure/subsystem) distinction.

I think we can say all of these are "behaviour executors":

- Subsystems
- Components
- Roles
- Actors

Marc: In ArchiMate's meta meta model, they are subtypes of the Structure Element.

Graham: I'm never entirely comfortable about such abstract supertypes. You might say Role and Actor are the most generic concepts, so then a component type is a role and a component instance is an actor. Or you might say Component type and instance are the most generic concepts. So then a role is a component type and an actor is a component instance.

To avoid heading down a path in which every concept is a subtype of another, we have to select only those supertypes which are most fundamental to the process of analysis and design; the ones we need to teach people about. I think the orthogonal relationship between process and component is fundamental to the process of analysis and design.

[See [3] for more discussion.]

Business architecture

Actors and Roles

ArchiMate's Actor	TOGAF's Actor
Organizational entity capable of performing behaviour.	A person, organization, or system that has a role that initiates or interacts with activities; for example, a sales representative who travels to visit customers. Actors may be internal or external to an organization.
	OR TOGAF's Organisation unit
	A self-contained unit of resources with line management responsibility, goals, objectives, and measures. Organizations may include external parties and business partner organizations.
ArchiMate's Role	TOGAF's Role
A named specific behaviour of a business actor participating in a particular context.	The usual or expected function of an actor, or the part somebody or something plays in a particular action or event. An actor may have a number of roles.

Marc: A Business Process can be assigned to a Role; similarly, a Role can be assigned to an Actor. A job title (Clerk) is a Role. A person (Smith) is an Actor who plays the Role. Several people can be appointed to the same job title.

Graham: The distinction is less clear in TOGAF. A TOGAF Actor has an attribute recording the number of FTEs that operate as that actor, which makes Actor sound like Role.

In any case, the Actor-Role distinction is unclear when there is only one actor for a role. E.g. is the BACS clearing system an actor, a role or both?

And in practice people commonly draw diagrams using the Actor symbol in place of Role, because they prefer the stick man symbol!

Marc: Actor being "external to an organisation" is odd, since the organisational entities are of course within the consideration of the enterprise architecture.

Graham: Just one more place where decomposition of entities destroys the meaning of an entity's definition! Traditionally actors are "external entities" outside the boundary of the system we model. TOGAF treats the enterprise as a system. Therefore, the actors are outside that the boundary. But the definition doesn't account for the fact that systems are nested. Obviously, applications are systems within the enterprise boundary. So some actors appear in the enterprise, but outside the applications.

Functions, Processes and Business Services

ArchiMate's Function	TOGAF's Function (aka Business Function)
Unit of internal behaviour that groups behaviour according to e.g. required skills, knowledge, resources, etc., and can be performed by a single role.	Delivers business capabilities closely aligned to an organization, but not necessarily explicitly governed by the organization. Also referred to as "business function".
	TOGAF's Capability
	A business-focused outcome that is delivered by the completion of one or more work packages. Using a capability-based planning approach, change activities can be sequenced and grouped in order to provide continuous and incremental business value.
	TOGAF's Process
ArchiMate's Organisation Service	TOGAF's Business Service
Externally visible functionality, meaningful to the environment and realized by business behaviour (process, function, interaction).	Supports business capabilities through an explicitly defined interface and is explicitly governed by an organization.

Graham: Notice a TOGAF Service is governed by an organisation unit. The meta model maps services also to functions. We discussed functions at great length in our previous paper. I think term *Business Function* means much the same in ArchiMate and TOGAF. But ArchiMate use the term Function outside of that, embracing what the TOGAF meta model calls Process.

Marc: Function is indeed one of the most troublesome concepts. And how does it relate to a TOGAF Capability?

Graham: The TOGAF meta model does not relate Function to Capability. Yet they are well-nigh indistinguishable. A Capability "is a" Function, a high-level business function. TOGAF 9's new chapter on Capability-Based Planning is not well integrated with other chapters in which capability means other things: e.g. A capability architecture is a project-level solution architecture.

I have posted some slides "What is a Function" at [7]. They relate Function to Capability to Organisation Unit to Service in the ways I infer from TOGAF.

Products and Contracts

ArchiMate's Product	TOGAF's Product
a coherent collection of services, accompanied by a contract/set of agreements, which is offered as a whole to (internal or external) customers.	Output generated by the business. The business product of the execution of a process.
AND/OR ArchiMate's Business Object	
a unit of information that has relevance from a business perspective.	
ArchiMate's Contract	TOGAF's Contract
specification of an agreement that specifies the rights and obligations associated with a product.	An agreement between a service consumer and a service provider that establishes functional and non-functional parameters for interaction.

Graham: One Contract is for a product; the other is for a Service.

Marc: True, but an ArchiMate product consisting of a single service would do the same thing.

Graham: I am unclear in this area. I don't understand what our two standards mean to say about the multiplicity of association relationships between Service, Interface, and Contract, and about how they relate to all the other entities they might be attached to.

Marc: ArchiMate is not very strict about multiplicity, to grant architects some leeway in its use.

Graham: I suggested earlier that we might do better to separate as many as six concepts in

the area of “Interfaces”.

Data architecture

ArchiMate’s Data Object	TOGAF’s Data Entity
A coherent, self-contained piece of information, e.g. customer record	An encapsulation of data that is recognized by a business domain expert as a thing. Logical data entities can be tied to applications, repositories, and services and may be structured according to implementation considerations.
	TOGAF’s Logical Data Component
	A boundary zone that encapsulates related data entities to form a logical location to be held; for example, external procurement information.
	TOGAF’s Physical Data Component
	A boundary zone that encapsulates related data entities to form a physical location to be held. For example, a purchase order business object, comprising purchase order header and item business object nodes.

Marc: ArchiMate uses data objects to model not only entities, but larger data sets and what TOGAF calls Logical Data Components.

Graham: Data sets and data components seem different to me. A TOGAF data entity covers persistent data (in a data store) and transient data (in a data flow). It defines data content rather than a data container.

I think a TOGAF data component (despite the purchase order example given) is best understood to be data container rather than data content. A data component represents a locatable container of persistent data – to which one or more applications can make a connection. A data component might contain a small data set, perhaps only one entity. The important thing is not the size, but that it exists and is accessible in a location.

Marc: For example, if we are interested in modelling Application Services that provide access to the data, using an Application Component (and perhaps an Application Function) might be called for.

Graham: Yes if there is an Application Component that encapsulates the data store in the baseline architecture, or you want to define one in the target architecture.

I note also that neither ArchiMate nor TOGAF give due prominence to event, data flow or message. But I’d rather leave “event” out for now. I argue that events (in behaviour models) are as important as entities (in structural models), and we always need to model from both perspectives. But once I start on that topic - it will turn into paper 5.

Marc: ArchiMate does have the Business Event concept and the Flow relation. But if we really want to use the power of events, we will also have to look at business rules for processing events. And business rules are neither addressed by ArchiMate nor by TOGAF. Paper 6?

Graham: My interest in entity-event modelling (from 1985 to 1995) was all about how to model business rules. Another time...

Applications architecture

ArchiMate’s Application Component	TOGAF’s Application Component
A modular, deployable, and replaceable part of a	An encapsulation of application functionality align-

system that encapsulates its contents and exposes its functionality through a set of interfaces.	ned to implementation structure.
	TOGAF's Logical Application Component
	An encapsulation of application functionality that is independent of a particular implementation. For example, the classification of all purchase request processing applications implemented in an enterprise.
	TOGAF's Physical Application Component
	An application, application module, application service, or other deployable component of functionality. For example, a configured and deployed instance of a Commercial Off-The-Shelf (COTS) Enterprise Resource Planning (ERP) supply chain management application.
ArchiMate's Application Service	TOGAF's Information System Service
Externally visible unit of functionality of an application component, exposed through well-defined interfaces, and meaningful to the environment.	The automated elements of a business service. An information system service may deliver or support part or all of one or more business services.

Graham: ArchiMate speaks of Application Components, Application Services, Application Interfaces and Application Functions.

TOGAF authors discuss related (not identical) architectural entities: Applications (in a portfolio thereof), Application Components (logical and physical), Application Services (aka Information Systems Services) and Application Functionality (ugly plural of Functions). Analysis suggests TOGAF uses Application Function to mean a process (or use case) rather than a component.

Am I allowed to disagree with the TOGAF meta model? I am tempted to do so in several places, and must do it here. It cannot be true in general that the physical application components defined in phases E and F are deployable instances of applications. They are at best application types, rather than instances. And even then, TOGAF tells us the "physical elements in an enterprise architecture may still be considerably abstracted from solution architecture, design, or implementation views".

Turning to another question: It is all too easy keep a class entertained for 30 minutes discussing the question: What is an application? Do you regard System Software as a kind of Application?

Marc: No, it is the software infrastructure on which applications run. Typical examples would be operating systems, DMBS's, messaging middleware, etc.

Graham: Which is Lotus Notes? Internet Explorer?

Marc: There is a trend for applications to become infrastructure over time. I would probably model both as system software.

Graham: OK, though infrastructure architects call all the software they deploy Applications. They also regard non-functional requirements as functional.

Marc: I know. I am not really partial to the term "non-functional", since it somehow sounds less important - which it isn't to architects of course.

Technology architecture

ArchiMate's Node	TOGAF's Technology Component
A computational resource upon which artifacts may be deployed for execution	An encapsulation of technology infrastructure that represents a class of technology product or specific technology product.
	TOGAF's Logical Technology Component
	An encapsulation of technology infrastructure that is independent of a particular product. A class of technology product
ArchiMate's Device	TOGAF's Physical Technology Component
Physical computational resource upon which artifacts may be deployed for execution	A specific technology infrastructure product or technology infrastructure product instance. For example, a particular product version of a Commercial Off-The-Shelf (COTS) solution, or a specific brand and version of server.
System Software	
A software environment for specific types of components and objects that are deployed on it in the form of artifacts	
ArchiMate's Artifact	
physical piece of information that is used or produced in a software development process, or by deployment and operation of a system (an executable, passive file, database table, etc.).	
ArchiMate's Infrastructure Service	TOGAF's Platform Service
Externally visible functionality of a node, exposed through well-defined interfaces, and meaningful to the environment.	A technical capability required to provide enabling infrastructure that supports the delivery of applications.

Graham: ArchiMate's Node (similar to the UML Node) is a *super type* of Device or System Software). But a TOGAF Logical Technology Component is an *idealised* component of the infrastructure hardware or software, meaning it is logical rather than physical.

Marc: So why wouldn't a Logical Technology Component match a Node, which is more abstract than Device and System Software?

Graham: Because it is more abstract in a different way. An ArchiMate node is abstraction by *generalisation* from two physical specialisations; so it is still physical. A TOGAF logical technology component is an abstraction by *idealisation* from physical technology components (the relationship could be one logical to one physical).

Marc: A subtle difference, but valid.

Graham: Whoever manages to align the ArchiMate and TOGAF meta models will have to detect and address such subtle differences.

From Node to Device

Graham: Again, my concern is to minimise ambiguity in my and students' minds about when and where they should use the ArchiMate box symbols. I'd like to press you about the distinctions between Node, Device, System Software and Application. I gather Node is a platform for applications and data

Marc: A Node is an abstraction of Device and System Software. It is "a computational resource upon which artifacts may be deployed for execution."

Graham: There are several kinds of "abstraction" and they are entangled with each other. You're saying Node is a *generalisation* of hardware or software platform. So I can use it in place of either specialisation, where I want to avoid being that specific. Are you relaxed about mixing levels of generalisation (Node and Device) in the same diagram?

Marc: I don't mind mixing both levels. I often draw Nodes composed of one or more Devices on which System Software is deployed. I could always use Node, but I tend use Device where the server hardware is named. A Device is a physical computational resource.

Graham: Ah! You may use Node in place of Device, not because you want the reader to be unconscious of whether it is a hardware or software, but rather because you want to be less vendor-specific. You're saying here the Node is an *idealisation* of Device.

Marc: I meant that using a vendor or product name is very specific and 'physical', since you denote the actual 'thing' instead of a more generic computational resource. It would be reasonable, when refining a model by adding physical product names, to use Devices instead of Nodes.

Graham: So when I (as designer) come to annotate your Node box with an IBM product number, I should change the box to physical Device. And when I decide to virtualise that Device, I should change the box symbol to System Software.

(Perhaps virtualisation will eventually remove the need for us the model hardware in an EA method at all.)

From Network to Device

Marc: The Network box symbol is used to model the network connections, routers, bridges and switches in the physical communication infrastructure.

Graham: I think of those as Nodes or Devices.

Marc: I don't use those boxes because network equipment isn't 'computational' in the strict sense. You cannot deploy a piece of software on a switch.

Graham: Surely routers, bridge and switches have network software deployed on them?

Marc: Yes, but from the user's perspective, they are 'closed' boxes - not generic computational resources that can be assigned all kinds of duties. However, the distinction is fuzzy. The border between Network and Device can be difficult to draw. I might model Firewalls as Devices (or Nodes), since they are relatively computational. But enterprise architects rarely model at the level of routers and switches. Mostly, they just show that a Node connects to a network (often a cloud).

Artifacts

Graham: TOGAF does not expect enterprise architecture to get down to the level of deployable artifacts. Again, TOGAF tells us the "physical elements in an enterprise architecture may still be considerably abstracted from solution architecture, design, or implementation views". It expects solution architects to get down to that level in phase G, after the TOGAF-level enterprise architecture has been completed.

Concluding remarks

Our analysis here is not complete, and we aren't planning to complete it. We think that is either somebody else's job, or deserving of a commercial contract! However, we do invite the TOGAF and ArchiMate working groups of The Open Group to take up this analysis and use it as input to improving the alignment and consistency of both methods. We trust we have shed some light on the sometimes subtle and abstract ideas behind these approaches and we hope that users of both TOGAF and ArchiMate can profit from this analysis.

References

- [1] Graham Berrisford & Marc Lankhorst, "Using ArchiMate with an Architecture Method", *Via Nova Architectura*, June 2009. <http://www.via-nova-architectura.org/magazine/magazine/using-archimate-with-an-architecture-method.html>
- [2] Graham Berrisford & Marc Lankhorst, "Using ArchiMate with TOGAF – Part 1: Answers to nine general questions about methods", *Via Nova Architectura*, June 2009. <http://www.via-nova-architectura.org/artikelen/tijdschrift/using-archimate-with->

togaf.html

- [3] Graham Berrisford & Marc Lankhorst, "Structural and behavioural views – and the ambiguity of "Function", *Via Nova Architectura*, July 2009. <http://www.via-nova-architectura.org/artikelen/tijdschrift/structural-and-behavioural-views-and-the-ambiguity-of-function.html>
- [4] The Open Group, *ArchiMate® 1.0 Specification*. Van Haren Publishing, 2009. <http://www.opengroup.org/archimate>, <http://www.opengroup.org/bookstore/catalog/c091.htm>,
- [5] The Open Group, *TOGAF™ Version 9*. Van Haren Publishing, 2009. <http://www.opengroup.org/togaf>
- [6] Marc Lankhorst, Hans van Drunen, "Enterprise Architecture Development and Modelling – Combining TOGAF and ArchiMate", *Via Nova Architectura*, 21 March 2007. <http://www.via-nova-architectura.org/magazine/magazine/enterprise-architecture-development-and-mode.html>
- [7] Graham Berrisford, Papers accessible at <http://avancier.co.uk>, in particular:
"What is a Function?"
"Architecture Meta Models"
Papers on "Abstraction" in the "Library".
- [8] British Computer Society, *Reference model for ISEB certificates in enterprise and solution architecture (v11.3)*, BCS, 2009. <http://www.bcs.org/upload/pdf/reference-model-enterprise-solution-architecture.pdf>

Copyright

This paper is edited from conversations between Marc Lankhorst (group leader Service Architectures at Novay and a founder and teacher of ArchiMate® [4]) and Graham Berrisford (a TOGAF™ 9 [5] instructor for Architecting-the-Enterprise and author of the papers in [7]).

Creative Commons Attribution-No Derivative Works Licence 2.0

No Derivative Works: You may copy, distribute, display only complete and verbatim copies of this document, not derivative works based upon it.

Attribution: You may copy, distribute and display this copyrighted work only if you clearly credit the authors Graham Berrisford of Avancier Limited and Marc Lankhorst of Novay before the start and include this footnote at the end.

For more information about the licence, see <http://creativecommons.org>

ArchiMate® and TOGAF™ are registered trademarks of The Open Group.

Graham Berrisford

graham.berrisford@architecting-the-enterprise.com

Marc Lankhorst

marc.lankhorst@novay.nl