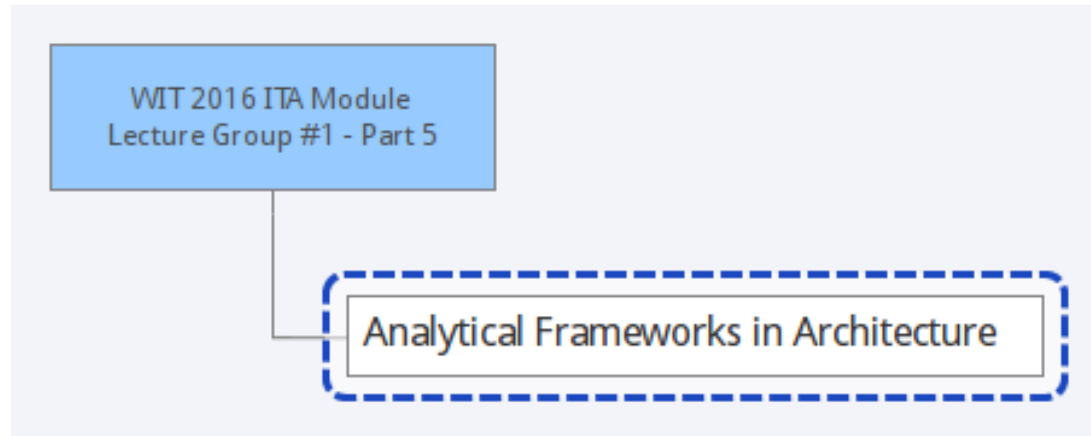

WIT 2016 ITA Module

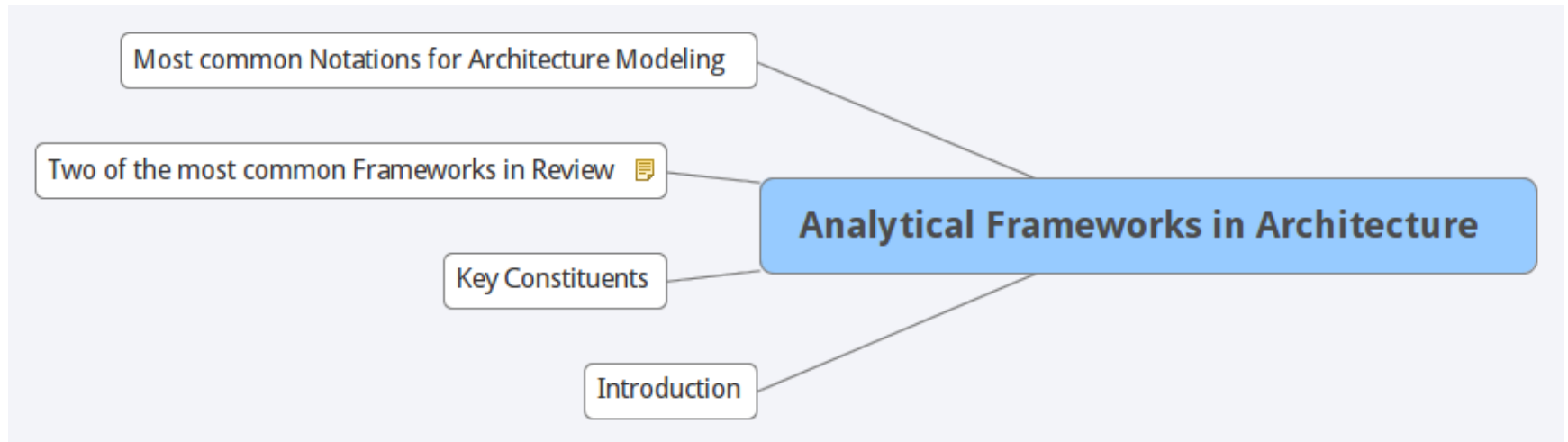
Lecture Group #1 - Part 5 Architecture Frameworks



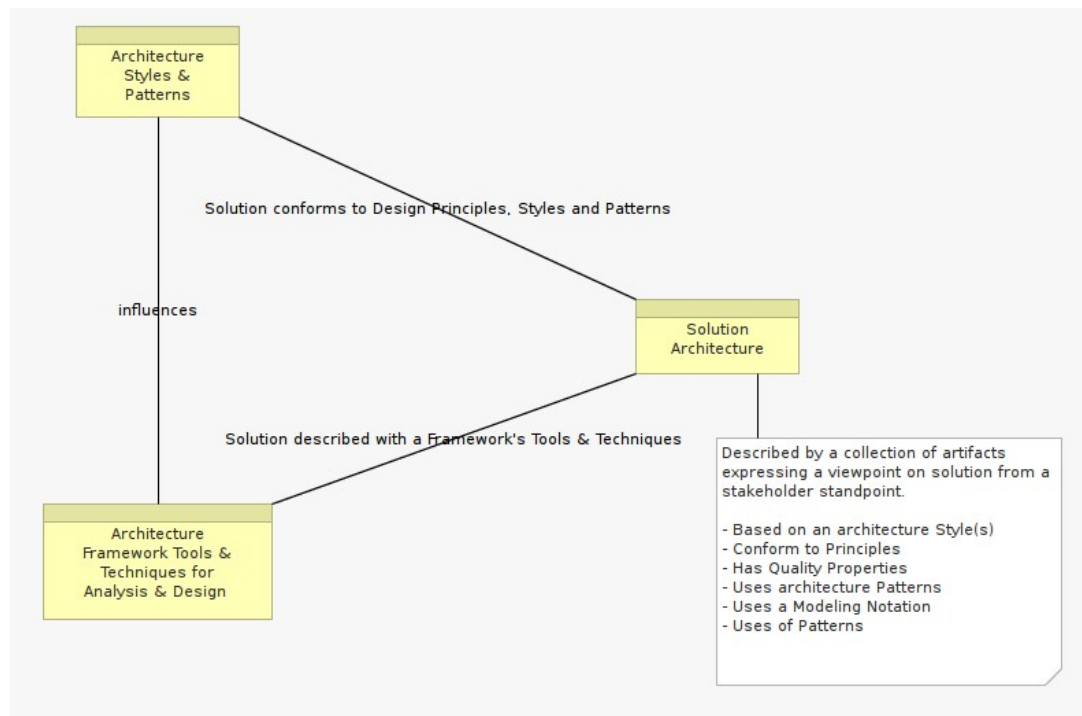
Lecture Group #1 - Part 5



Analytical Frameworks in Architecture



What is an Analytical Framework?



Rationale

It is difficult to compare and discuss architecture options without a common frame of reference providing the basic semantics of a common language.

Any Transfer of knowledge requires dialog.

Dialog requires a common language to describe concepts.

Architecture Frameworks provide analytical Tools and Techniques to help Architects understand the "Problem to Solve" and Design answering Solutions.

Architecture Frameworks provide a common base of reference for architects, making it easy to share and transfer best practices.



Definition

An Analytical Framework is a skeletal support used as the basis for something being constructed; a structure for supporting (or enclosing) something else.

It can loosely be defined as a set of rules and practices that constitute a way of viewing (i.e. analysing) the reality of the Business Enterprise.

Frameworks recommend standardized analysis techniques, tools, terminology, methods, standards and design protocols.

Analytical Frameworks focus on the problematic of:

- (1.) Management of complexity, ...and cost implications,
- (2.) Ease of change, ...and cost implications,
- (3.) Alignment of Business & IT expectations, and cost implications.



Purpose

To propose a uniformed approach for the discovery, identification, sequencing, planning, implementation, integration of the "building blocks" of a solution.

- (1.) How Investigate a Problem, thinking problem complexity,
- (2.) How to create a Solution (i.e. using classifications, viewpoints)
- (3.) How to manage change, technical debt, encouraging re-use, discouraging re-work and avoiding waste.

To foster consistency of skill across Architects

...in the use of standard deliverables issued from standardized analysis and design approaches

...avoiding the creation of "stovepipe" Solutions, that do not interact with or build on other applications, are difficult to enhance, creating "entropy".



Use of Framework

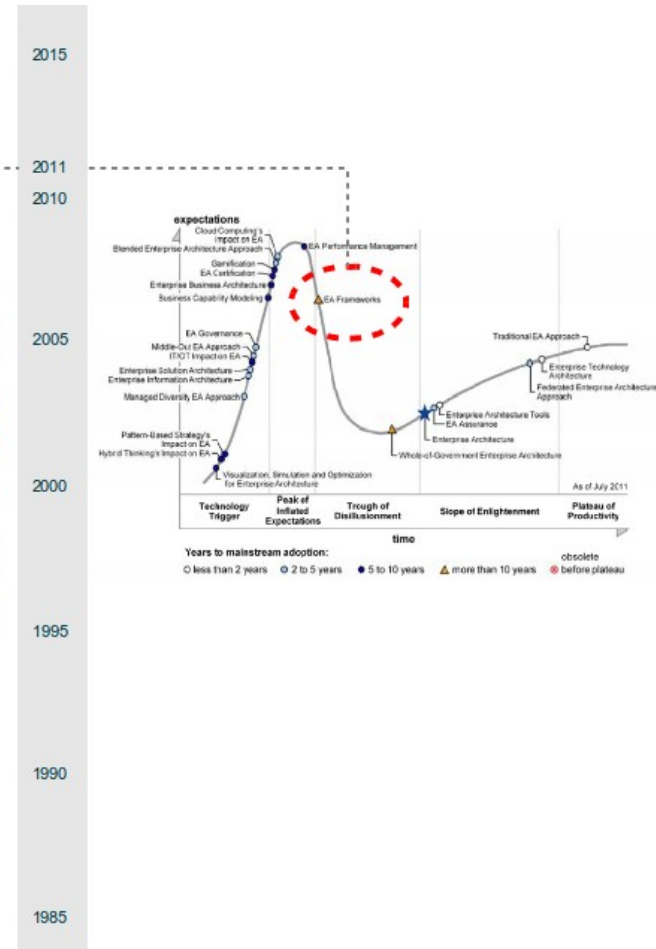
On its own, and while extremely useful, an Analytical Framework simply embodies a set of conceptual guidelines.

Architects adjust its aspects to the needs and characteristics of the problem and solution design.



State of the Art in 2016

- (2011) ISO/IEC/IEEE 42010:2011
- (2011) TOGAF™ 9.1
- (2011) Zachman Framework v3.0
- (2011) RM-ODP ISO-EntVP
- (2010) FEF v1.0
- (2009) TOGAF™ 9.0
- (2006) IEEE 1471 is an ISO Standard
- (2005) MetaGroup/Gartner AF Research
- (2004) Enterprise Unified Process (EUP) v5.0
- (2003) E-VAP™
- (2003) TOGAF™ 8.0 (Enterprise Edition)
- (2001) TOGAF™ 7.0 (Technical Edition)
- (2000) IEEE 1471 AF Standard
- (1999) Visual Architecture Process™ (VAP) v1.0
- (1999) Federal Enterprise Architecture Framework (FEAF)
- (1998) Rational Unified Process (RUP) v5.0
- (1998) Zachman Framework v2.0
- (1996) Rational Objectory Process (ROP) v4.0
- (1996) RM-ODP
- (1992) Zachman Framework v1.0
- (1988) Objectory v1.0
- (1987) Zachman Framework for IS Architecture



State of the Art in 2016

There are around 50+ named Architecture Frameworks in existence, tackling a variety of concerns, but few are truly distinct from one another.

Some Frameworks originate from IT and evolved to encompass the Enterprise.

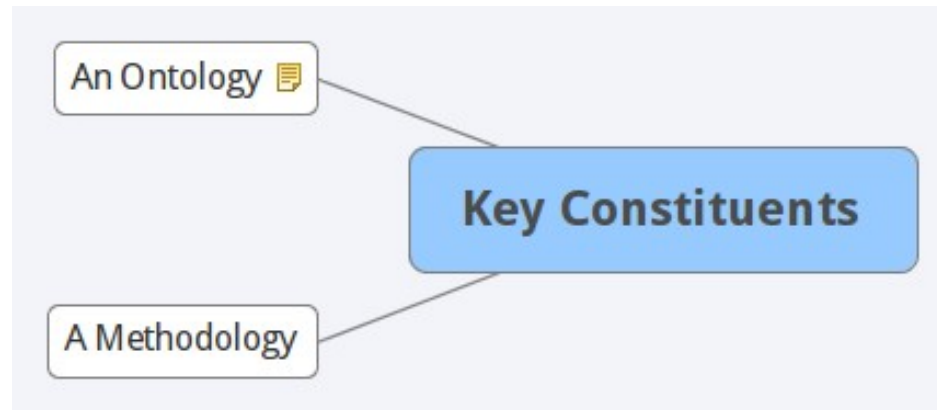
Some Frameworks are originally designed to model the Enterprise as a System.

Some Frameworks extend the modeling to Enterprise Planning.

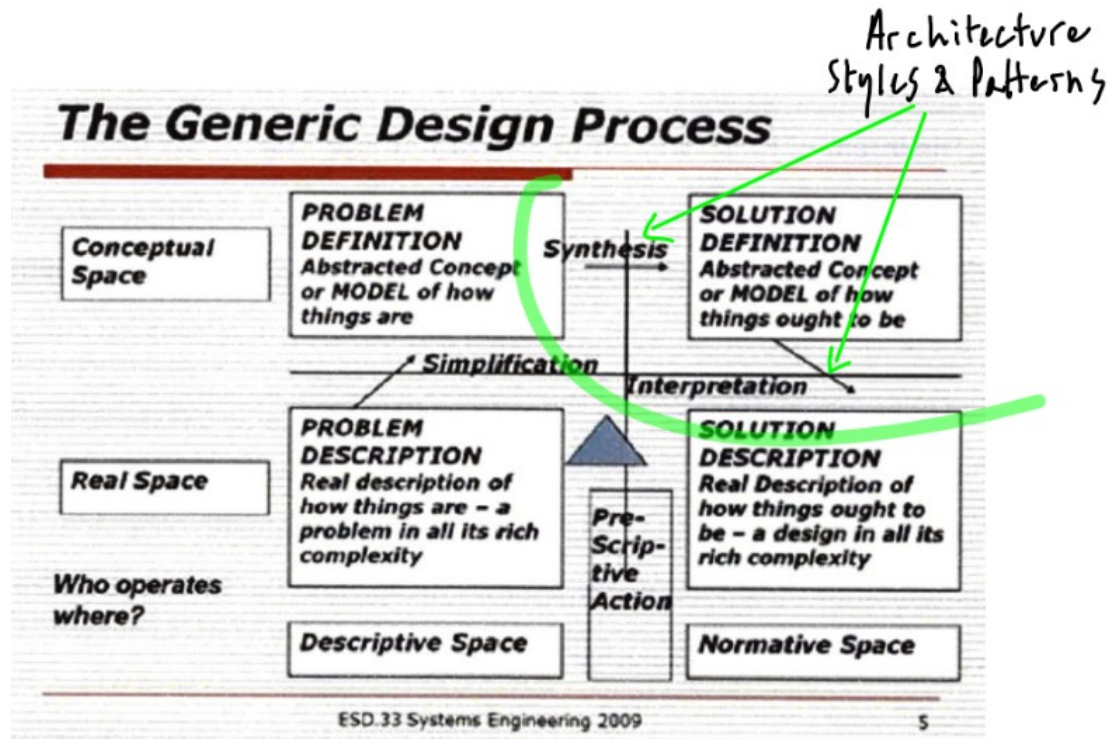
Some Frameworks are anchored in Software Engineering & Architecture, and require adjustment to the Enterprise.



Key Constituents



A Methodology



The selection & use of Architecture Styles & Patterns often take places during "Synthesis" and "Interpretation" steps.

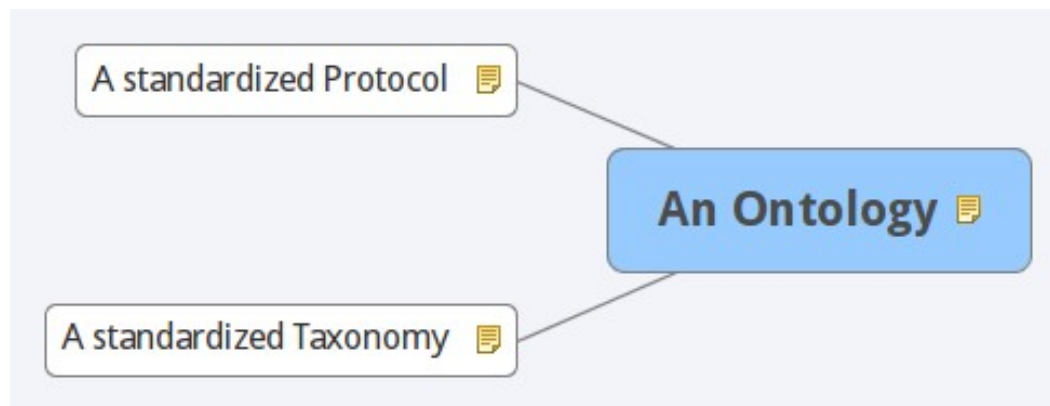


An Ontology

The Enterprise is seen as a multi-dimensional space and therefore need to be assessed under a set of different but interrelated angles.

It is not possible to capture the functional features and quality properties of a complex system in a single comprehensible model that is understandable by and of value to all stakeholders.

For this reason Architectural Frameworks propose a way of classifying descriptive deliverables/artifacts using different dimensions, aimed at specific stakeholders.



The "Primitives" of Architectural representation



An Enterprise is a complex System and as such is composed of many Domain Specialists.

Specialists have a perspective on things, a natural appreciation of a given subject of their field).













Viewpoints help to characterize and distinguish Architectural representations, and make them understandable to Specialists.

The intersection of 2 lines qualifies the simplest expression of an architectural view that addresses a precise concern.



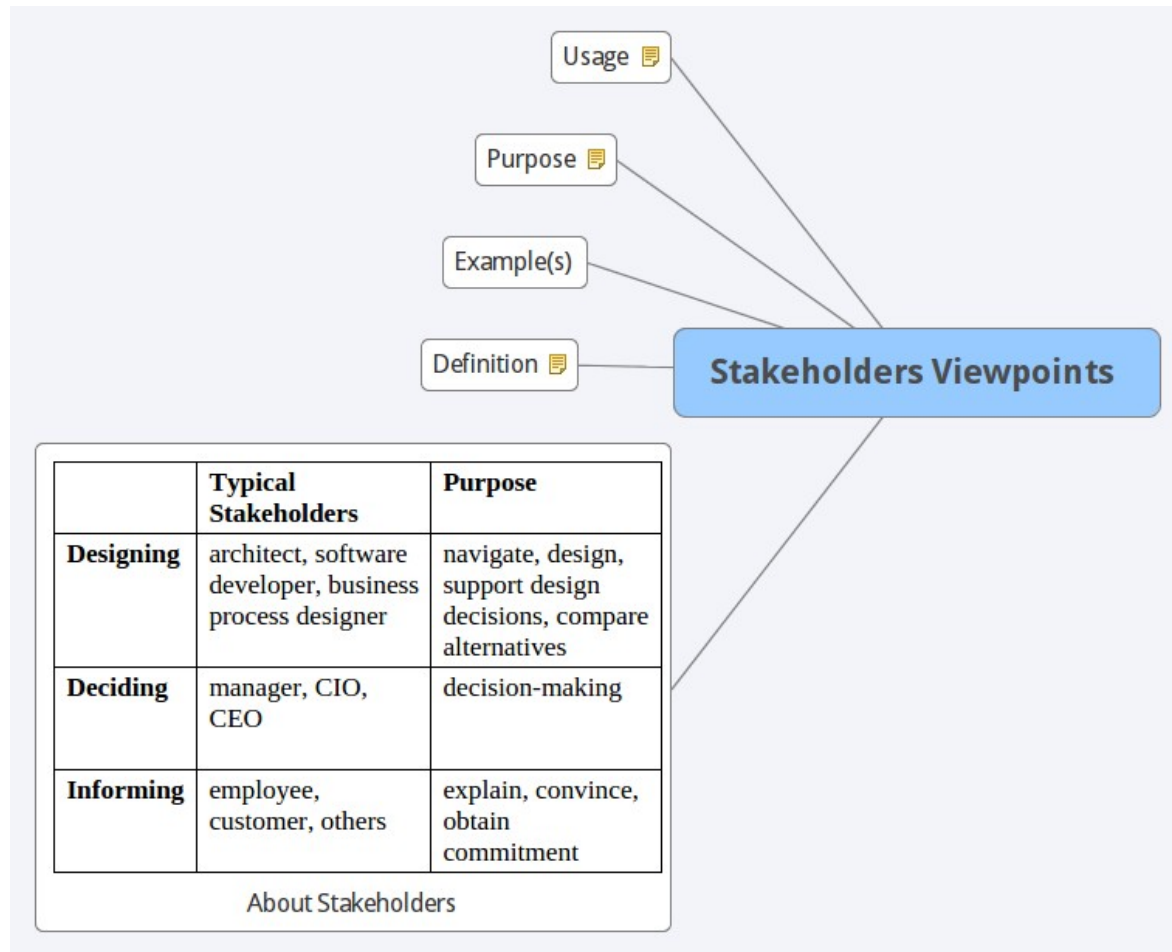
Example: Zachman Framework

Screenshot taken from ArchiMetric / VisualParadigm 2016

	WHAT	HOW	+	WHO	WHEN	+	
SCOPE CONTEXTS	Inventory Identification  Inventory Types	Process Identification  Process Types		Organization Identification  Organization Types	Timing Identification  Timing Types		STRATEGISTS AS THEORISTS
BUSINESS CONCEPTS	Inventory Definition  Business Entity Business Relationship	Process Definition  Diagrams 3 Terms 5 Rules 3 Business Transform Business Input		Organization Definition  Business Role Business Work	Timing Definition  Business Cycle Business Moment		EXECUTIVE LEADERS AS OWNERS
TECHNOLOGY PHYSICS	Inventory Specification  Technology Entity Technology Relationship	Process Specification  Technology Transform Technology Input		Organization Specification  Technology Role Technology Work	Timing Specification  Technology Cycle Technology Moment		ENGINEERS AS BUILDERS
	INVENTORY SETS	PROCESS TRANSFORMATIONS	+	ORGANIZATION GROUPS	TIMING PERIODS	+	



Stakeholders Viewpoints



Definition

A stakeholder Viewpoint determines the language (including notations, model, or product types) to be used to represent the architecture from the standpoint of a stakeholder.

Viewpoints express how to model a "view" of the solution architecture that speaks to stakeholder.

Viewpoints are the building blocks of a library of Templates that can be used off the shelf to guide the creation of an Architecture Solution (ex. TOGAF 9.1/Archimate 2.1 Viewpoints).



Purpose

To give architect a key (or legend) to understanding the notations in views of that type.

Viewpoints are an important way of bringing much-needed structure and consistency to what was in the past a fairly unstructured activity.

Viewpoints are defined in a uniform manner can be documented, put on the shelf, reused, and improved upon across the community of architects.



Usage

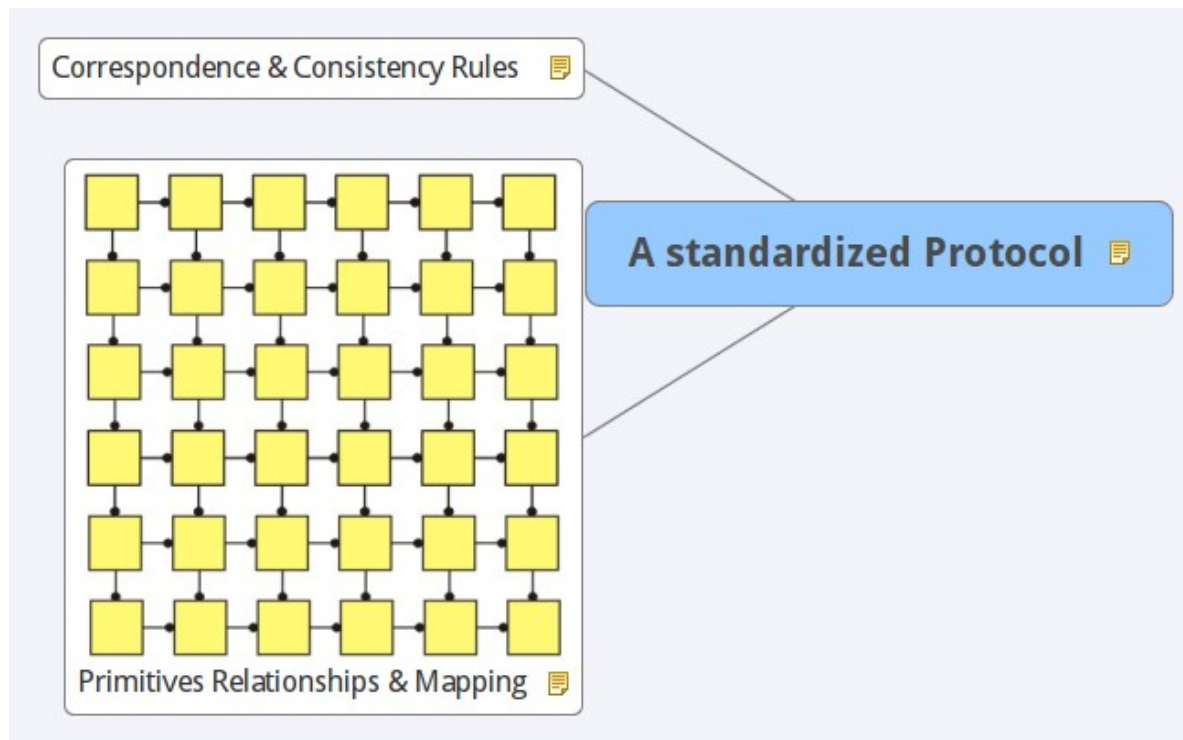
The identification and selection of stakeholders, concerns, and viewpoints, and the construction of model artifacts, is the responsibility of the architect, in association with the stakeholders.

Frameworks do not dictate to Architects which Viewpoints to use, because there is no consensus on which aspects of the problem or solution are important, and because systems vary widely in purpose.

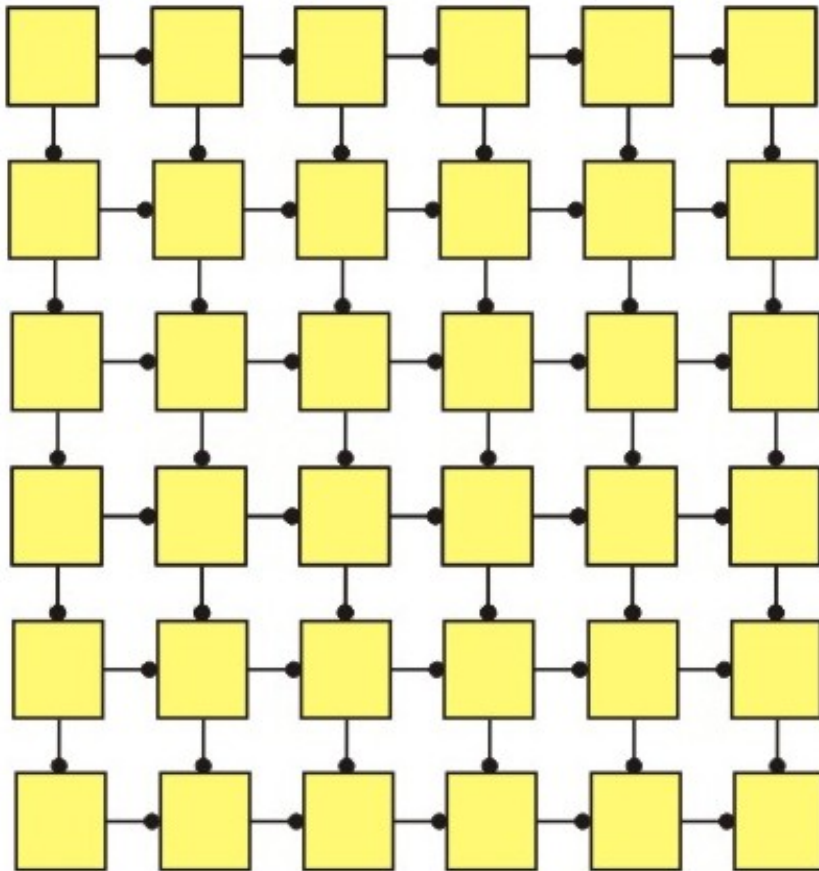


A standardized Protocol

Protocols address problems that typically arise whenever multiple Views are used to generate an Architectural Description.



Primitives Relationships & Mapping



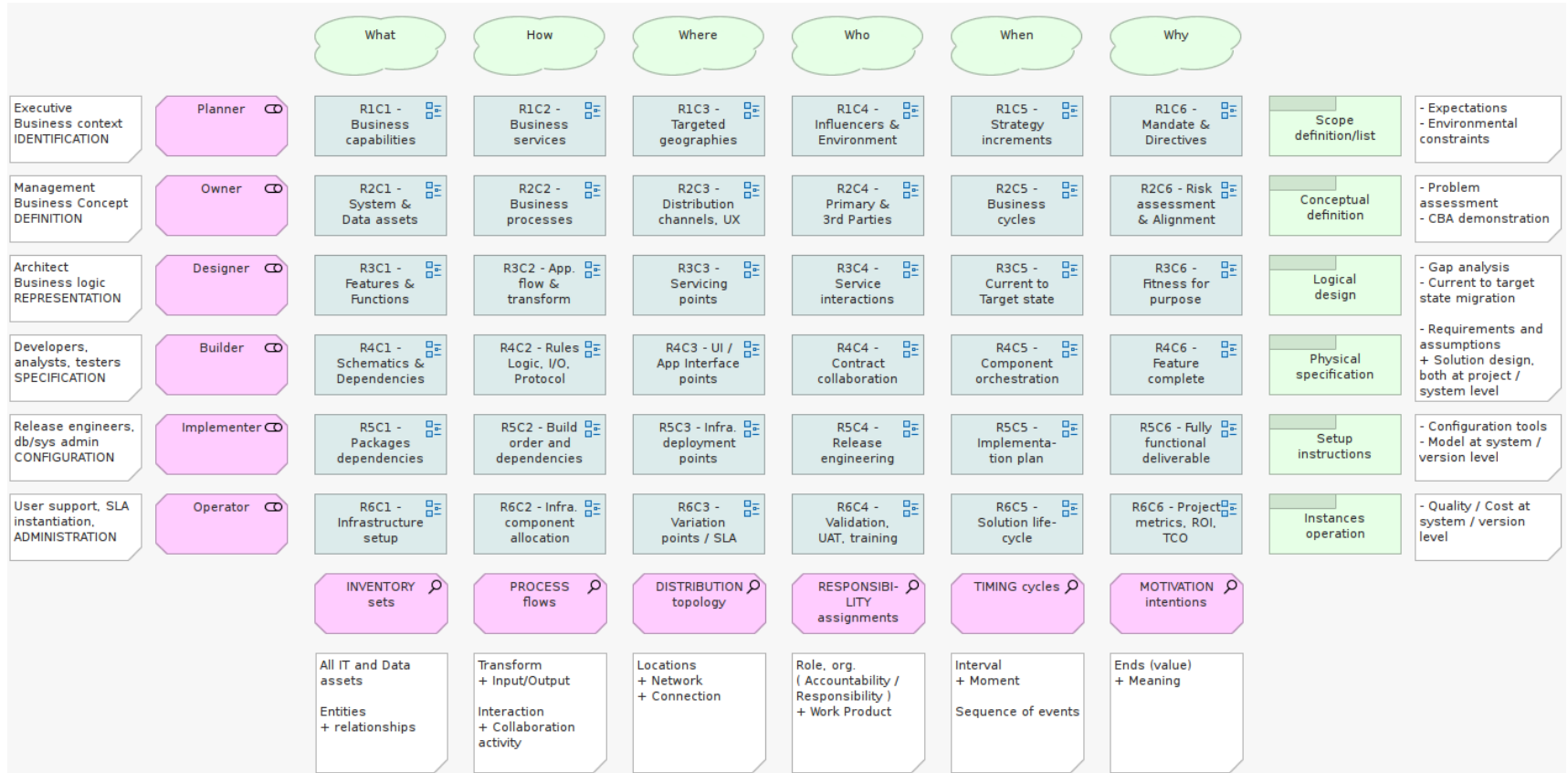
Taxonomies implicitly logical relationships between Entities.

Relationships between entities classified are one to many left to right and one to many top to bottom.

Relationships define "protocols" speaking to the order in which Entities should be (can be) created.

Arbitrary and more explicit relationships can be defined to superset the implicit two-dimensional structure.

Example of Framework Instantiation

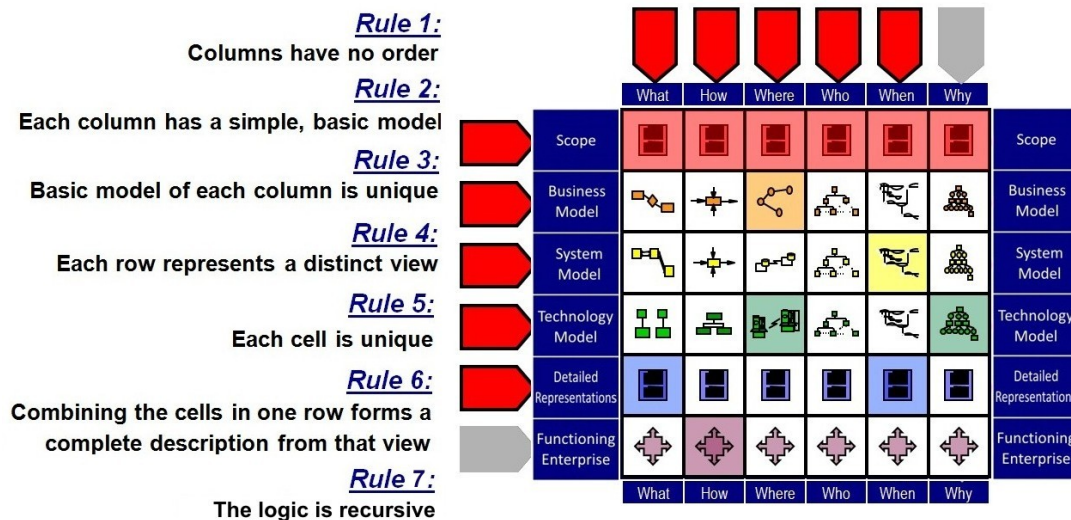


Correspondence & Consistency Rules

Correspondence Rules express relationships between elements within an architecture description to enforce architecture relations such as:

(1.) Composition, (2.) Dependency, (3.) Constraint and obligation, (4.) Traceability.

Consistency Rules enforce a common level of refinement and detail of specification across Entities.



Two of the most common Frameworks in Review

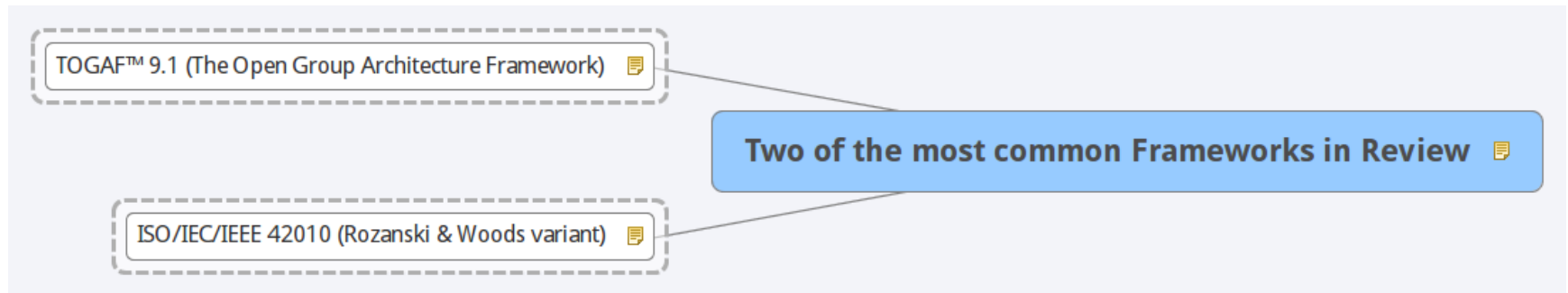
At a high-level, all Frameworks are composed of "building-blocks": (1.) a "Skeleton Structure", (2.) a "Nervous System", (3.) a "Tool-box".

They specify a widely tested and recognized set of "building blocks" to manage the Enterprise Architecture.

They propose a logical representation of the Enterprise.

They provide a set of practices to construct an Architecture.

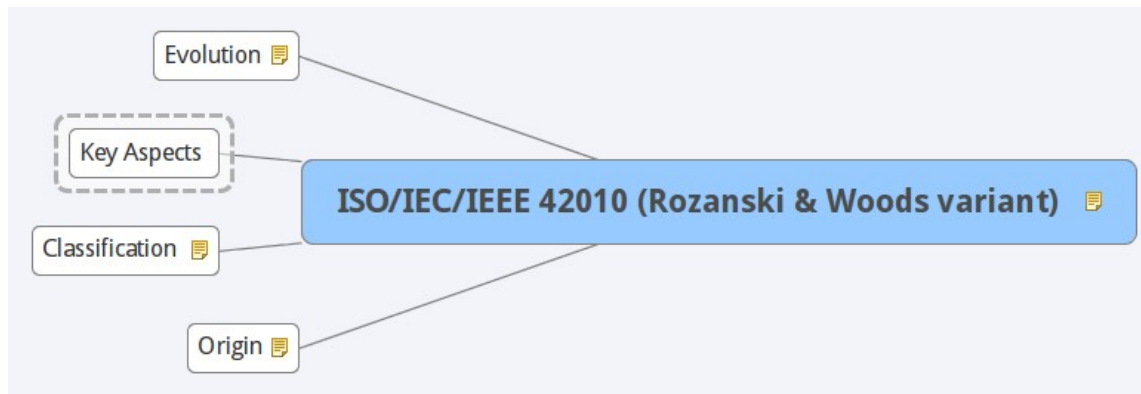
Some Frameworks are more prescriptive than others in the Contents, Methodologies and Tools they advise.



ISO/IEC/IEEE 42010 (Rozanski & Woods variant)

A taxonomy-centric Framework, based on IEEE 1471.

The Rozanski & Woods Framework shows how a modern, IEEE/ISO compliant EA framework can be implemented from the ISO/IEC/IEEE 42010 standard.



Origin

Originally defined in 2000 (as IEEE-1471) to provide a direction for incorporating Architectural "thinking" into IEEE standards.

IEEE 1471 was developed by the IEEE Architecture Working Group under the sponsorship of the IEEE Software Engineering Standards Committee.

The current Active Standard, published in 2011, is the result of a joint ISO and IEEE revision of the earlier IEEE Std 1471:2000, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.



Classification

IEEE-1471 is a taxonomy-centric architecture framework.

It proposes a set of Architecture Viewpoints to codifying conventions and common practices of Architecture Description.

It provides a framework and vocabulary for talking about the constituents of an Application Architecture.



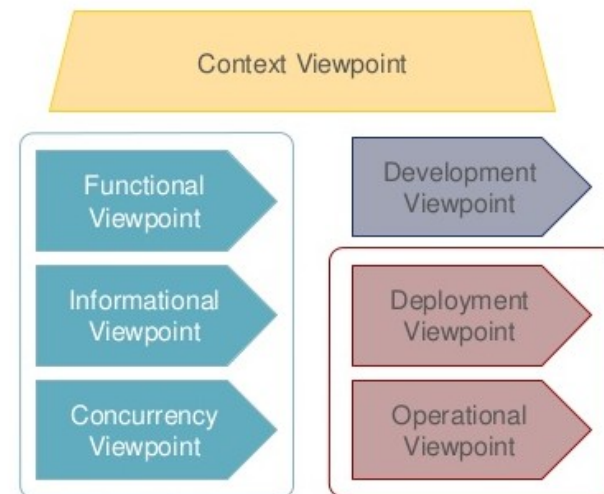
Ontology

Context Viewpoint: Business Management, Business Analysts

Functional, Informational, Concurrency Viewpoints: System Analysts, Database designers and System Designers.

Development Viewpoint: System and Software Engineers.

Deployment and Operational Viewpoints: Administrators, Release Engineers, Operations



Evolution

2000: The IEEE Standards Board approved IEEE 1471 for use.

2006: IEEE 1471 was adopted as an ISO standard.

2007: Publication of the Standard as ISO/IEC 42010:2007 - text identical to IEEE 1471:2000.

2011: ISO/IEC/IEEE 42010:2011 replaces both ISO/IEC 42010:2007 and IEEE Std 1471:2000.

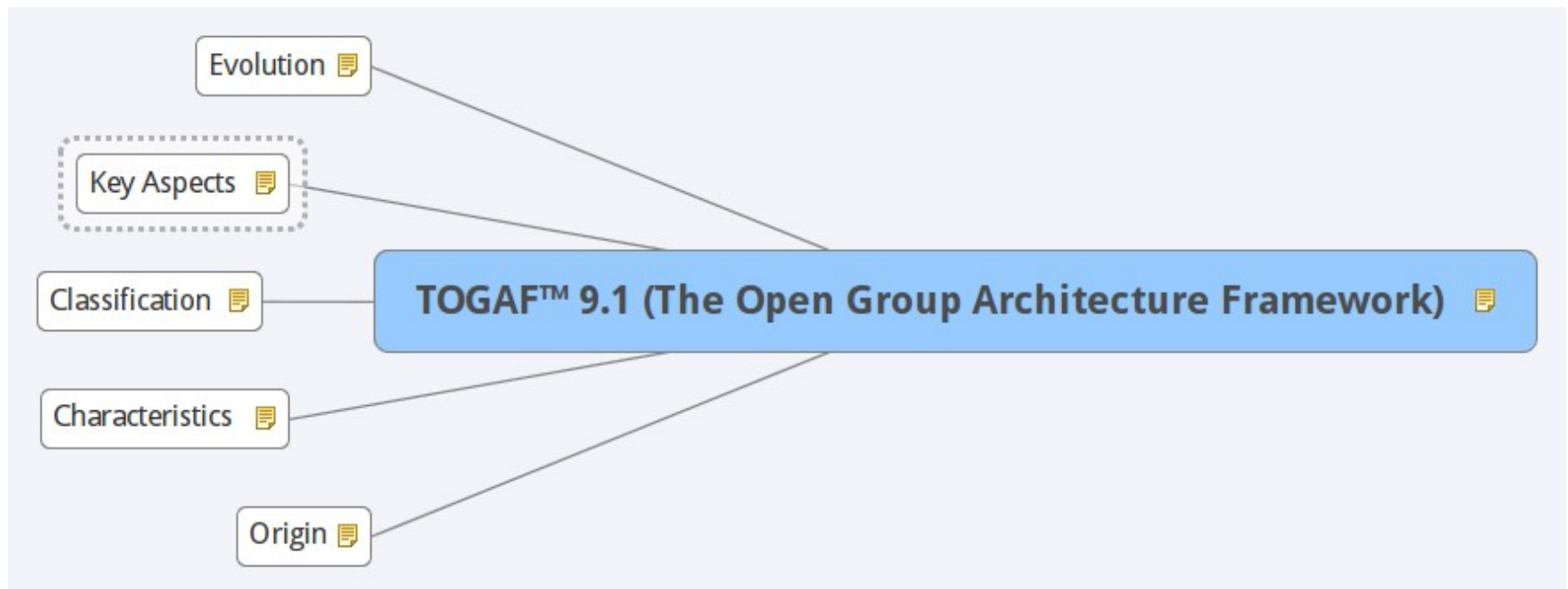


TOGAF™ 9.1

The Open Group Architecture Framework

A methodology-centric Framework.

The TOGAF Framework provides a comprehensive approach to the Planning, Design, Implementation, and Governance of an Architecture.



Origin

The Open Group Architectural Framework (TOGAF) was first developed in 1995 and is based on the Department of Defense's Technical Architecture Framework for Information Management.

The TOGAF framework definition focuses on mission critical business initiatives that use IT Systems as building blocks.

It has been developed by the Architecture Forum user group within The Open Group (CapGemini, HP, HSBC, IBM, NEC, SAP, Sun Microsystems).

It is vendor neutral, Technology neutral, Tool neutral. It is currently at revision 9.1.



Characteristics

A core and key element of TOGAF is the Architecture Development Method (ADM), which specifies a Process for creating Architectures.

TOGAF relies heavily on modularization, standardization and already existing, proven technologies and products.

TOGAF explains rules for developing good principles, rather than providing a set of architecture principles. Three levels of Principles support decision making across the entire enterprise, provide guidance of IT resources and support architecture principles for development and implementation.



Key Aspects

The "TOGAF Content Framework" provides a loose taxonomy to categorize Architectural primitives.

This taxonomy aims to drive consistency in the outputs from TOGAF.

It promotes reuse through consistency – a reusable "part" can be used in future architectures.

It promotes consistency between solution designs (and in turn the overall resulting architecture).



Ontology

To address the concerns of the following stakeholders...

Users, Planners, Business Management	Database Designers and Administrators, System Engineers	System and Software Engineers	Acquirers, Operators, Administrators, & Managers
---	---	----------------------------------	--

...the following viewpoints may be used to develop views of your solution architecture

Business Viewpoint	Data / Information Viewpoint	Application Viewpoint	Technology / Infrastructure Viewpoint
Business Function View	Data Entity View	Software Engineering View	Networked Computing/ Hardware View
Business Services View			
Business Process View			
Business Information View			
Business Locations View			Data Flow View (Organization Data Use)
Business Logistics View			
People View (Organization Chart)			
Workflow View	Processing View		
Usability View	Logical Data View	Software Distribution View	Cost View
Business Strategy and Goals View			
Business Objectives View			
Business Rules View			Standards View
Business Events View			
Business Performance View	System Engineering View		

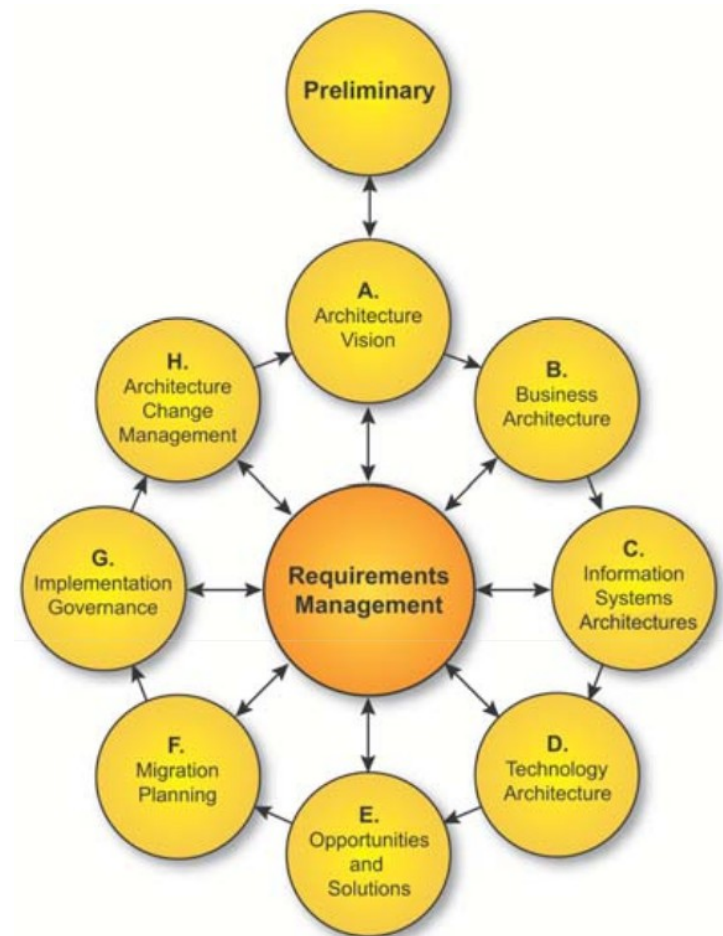


Methodology

The TOGAF 9.1 Methodology is referred as the Architecture Development Process (ADM).

It aims to be customizable for a given Organization and Project.

It is iterative over the whole process, between phases, and within phases.



Evolution

2001: TOGAF 7 ("Technical Edition") is published.

2003: TOGAF 8 ("Enterprise Edition") is published and updated regularly with new examples.

2009: TOGAF 9 includes: (1.) a new Content Meta-model that links the artifacts of TOGAF together, (2.) New Templates, (3.) Practices for Architecture scoping and segmentation, (4.) Practices for Business capability-based planning, (5.) Guidance on how to use TOGAF to develop Security Architectures and SOA.

2011: Maintenance release, TOGAF 9.1. is the current version of the Framework.



Common Notations for Application Architecture Modeling

1. The Unified Modeling Language standard
2. The ArchiMate® 2.1 Architecture Modeling standard



Unified Modeling Language (UML 2.x)

The UML standard is maintained by the Object Management Group (OMG).

It provides 14 types of diagrams divided into two categories: structure diagrams and behavior diagrams.

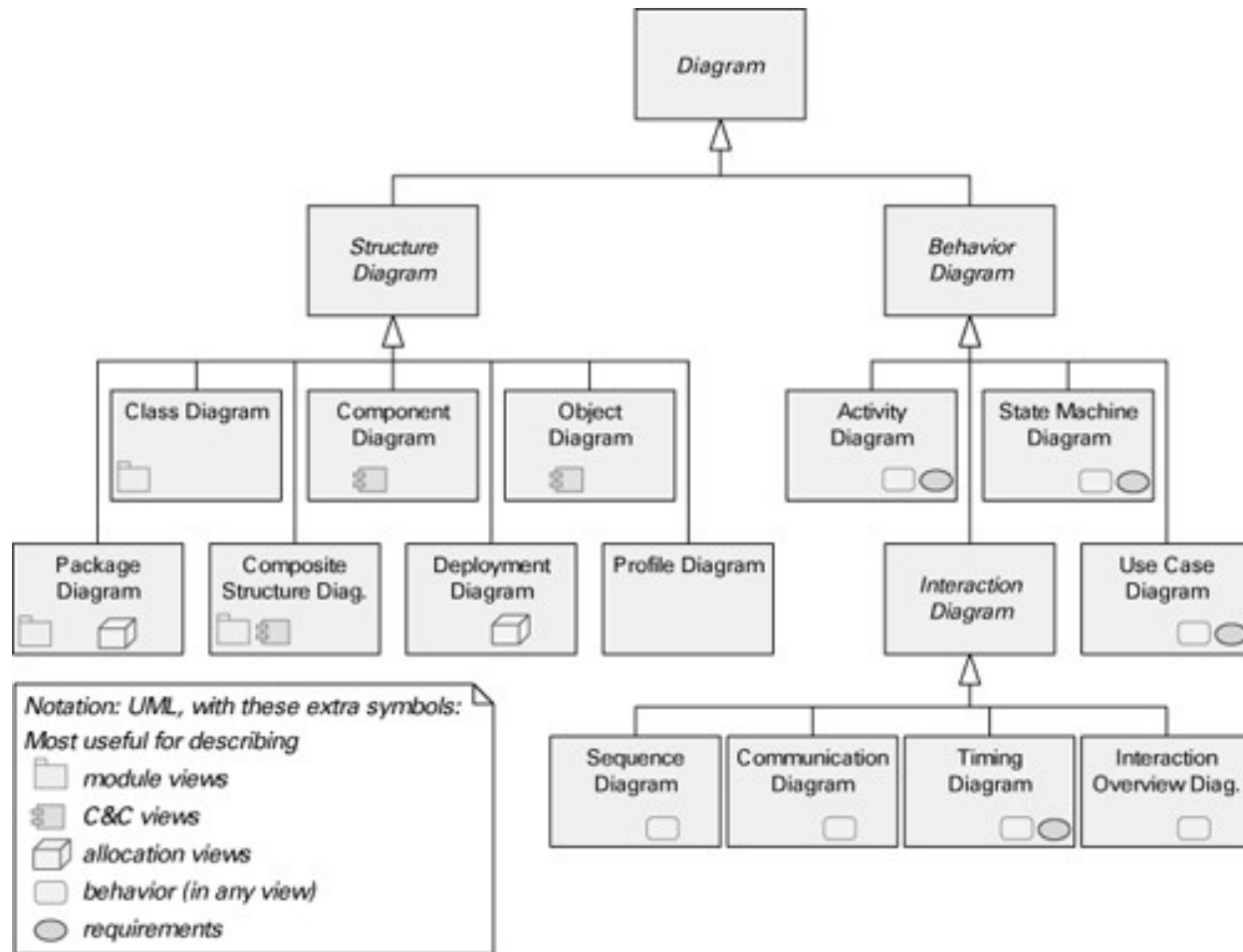
The meaning of any UML symbol can be specialized by using stereotypes.

A "Stereotype" is a domain-specific or technology-specific label shown within "angle brackets" that can be applied to existing UML elements to best describe an Architecture concept or relation.

UML is actively in practice today with strong tool support, especially in Manufacturing centric Industries (not so much in the non-manufacturing Enterprise).



UML 2.x Notation



ArchiMate® 2.x

ArchiMate is an open and independent modeling language for EA, created by the Open Group and fully aligned with the TOGAF EA Framework.

It is a Notation for describing, analyzing and visualizing relationships amongst Architectural Domains as defined in TOGAF.

Each Domain is specified by a Meta-Model, constraining the Diagrams that can be created, and allowing consistency of notation and re-use of Concept elements between Views.

ArchiMate also allows the connection of Models belonging to different "layers" (i.e. Business/Application/Data/Technology layers), hence helping an Architect to document View consistency.



Core Notation introduced in v1.0

	STRUCTURAL CONCEPTS		BEHAVIORAL CONCEPTS		INFORMATIONAL CONCEPTS					
BUSINESS	Business actor		Business role		Business process		Business service		Representation	
	Business collaboration		Business interface		Business function		Business event		Meaning	
	Location		Business object		Business interaction				Value	
APPLICATION	Application component		Application collaboration		Application function		Application interaction			
	Application interface		Data object		Application service					
TECHNOLOGY	Node		Device		Infrastructure function		Infrastructure service		Artifact	
	Network		System software							
	Communication path		Infrastructure interface							

