On the Zachman Framework

Introduction

- The human and computer activity systems of a large enterprise are complex.
- Comprehensive descriptions of those systems must also be large and complex.

- People need a taxonomy or classification scheme to help them organize system description artefacts.
- The classification scheme may be called a description or document or content framework.

- Here is it called schema.

Three candidate dimensions for a classification schema

- Candidate dimensions for a table mapping one dimension of business system architecture definition to another.

Composition	Generalisation	Idealisation
Coarse-grained composite	Universal	Concept
Mid-grained composite	Fairly generic	Logical Model
Fine-grained composite	Fairly specific	Physical Model
Elementary part	Uniquely configured	Physical Material
Decomposition	**Specialisation**	**Realisation**

E.g.

- A schema like this provides a two-dimensional index to descriptive artefacts. You can think of it as set of pigeon holes.

Generalisation / Composition	Universal	Fairly generic	Fairly specific	Uniquely configured
Coarse-grained composite				
Mid-grained composite				
Fine-grained composite				
Elementary parts				

A different set of pigeon holes…

- Mapping POLDAT (the six domains of change in the Catalyst methodology of CSC) to levels of composition.

Domains / Composition	Process	Organisation	Location	Data	Application	Technology
Coarse-grained composite						
Mid-grained composite						
Fine-grained composite						
Elementary parts						

A different set of pigeon holes…

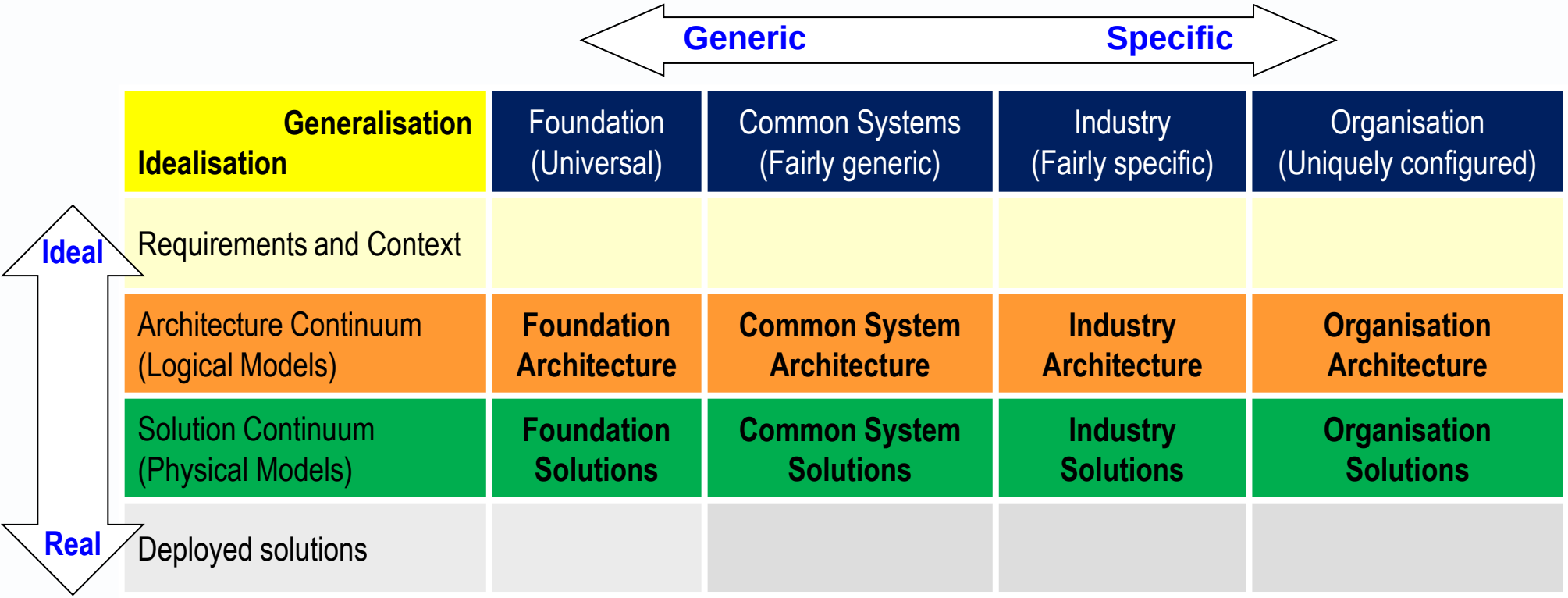
- Mapping POLDAT (the six domains of change in the Catalyst methodology of CSC) to levels of idealisation

Domains / Idealisation	Process	Organisation	Location	Data	Application	Technology
Conceptual						
Logical						
Physical						
Real						

TOGAF's "Enterprise Continuum"

Avancier

- This maps levels of idealisation to levels of generalisation.



Idealisation \ Generalisation	Foundation (Universal)	Common Systems (Fairly generic)	Industry (Fairly specific)	Organisation (Uniquely configured)
Requirements and Context				
Architecture Continuum (Logical Models)	**Foundation Architecture**	**Common System Architecture**	**Industry Architecture**	**Organisation Architecture**
Solution Continuum (Physical Models)	**Foundation Solutions**	**Common System Solutions**	**Industry Solutions**	**Organisation Solutions**
Deployed solutions				

- Architects can assign each description artefact to a cell of the schema, then use the schema as an index to find artefacts in a repository.

So, to the Zachman Framework

- A structure for classifying architecture description artefacts.
- Presented in 1987 as an “Information System Architecture Framework”, but since the mid 1990s has been called an EA Framework.

In 2008, the Zachman International web site quoted Zachman

- "The Zachman Framework is a schema - classifications that have been in use for literally thousands of years.

- The first is the fundamentals of communication found in the primitive interrogatives:
- **What, How, When, Who, Where, and Why**.
- It is the integration of answers to these questions that enables the comprehensive, composite description of complex ideas.

- The second is derived from reification, the transformation of an abstract idea into an instantiation that was initially postulated by ancient Greek philosophers and is labeled in The Framework:
- **Identification, Definition, Representation, Specification, Configuration and Instantiation."**

So, in its purest form, Zachman's schema would be

- Map 5 levels of idealisation to 6 analysis questions

Columns show "the primitive interrogatives"

Idealisation	Question	What	How	Where	Who	When	Why
Ideal	Identification						
Conceptual	Definition						
Logical	Representation						
Physical	Specification						
	Configuration						
Real	Instantiation						

Rows show "reification - the transformation of an abstract idea into an instantiation"

Real

But the Zachman framework was long introduced as

Avancier

"A logical structure for classifying and organizing the descriptive representations of an Enterprise that are significant to managers and to developers of Enterprise systems."

- "It uses a grid of 6 basic **questions** asked of 5 **stakeholder groups**

Question / Idealisation	What	How	Where	Who	When	Why
Planner						
Owner						
Designer						
Builder						
Subcontractor						
Operations						

- Zachman, along with most EA, is less concerned with operational systems at the bottom, more with the description and documentation above.

To illustrate what idealisation means

- This is an interpretation, not necessarily what Zachman would propose

Question / Idealisation	What	How	Where	Who	When	Why
Ideal						
Planner		External Requirements and Drivers Business Function Models				
Conceptual						
Owner			Business Activity Business Data Models			
Logical						
Designer			Logical Models Requirements Definition			
Physical						
Builder			Physical Models Solution Development			
Real						
Subcontractor			Code and data definition Deployable to computers			
Operations			Running systems Monitoring of systems			

1987: The Zachman Framework for IS Architecture - version 1

- Mapped the 6 questions to architectural elements
- Mapped the 5 levels of abstraction to stakeholders.

Zachman Framework v1			What	How	Where	Who	When	Why
Viewpoint	**Idealisation**	**Stakeholder**	Data	Function	Network	Org.	Schedule	Strategy
Scope	Contextual	Planner						
Enterprise	**Conceptual**	Owner						
System	**Logical**	Designer						
Technology	**Physical**	Builder						
Detailed	Out of context	Subcontractor						
	Functioning Enterprise							

2009: The Zachman Framework for EA (v 2)

Avancier

- Zachman grew uncomfortable about what he saw a misinterpretations.
- E.g. “What” is not only about data. So he changed that to “inventory sets”.
- And rows were relabelled to show “reification” of descriptive artefacts as things in operational systems

Zachman Framework v2			What	How	Where	Who	When	Why
Viewpoint	**Idealisation**	**Stakeholder**	Inventory sets	Process Transform’n	Network nodes	Org. groups	Time periods	Motivation reasons
Scope	Contexts	Strategists & theorists						
Business	**Concepts**	Enterprise leaders & owners						
System	**Logic**	Architects & designers						
Technology	**Physics**	Engineers & builders						
Component	Assemblies	Technicians & implementers						
Operations	**Instance classes**	Workers & participants						

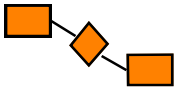
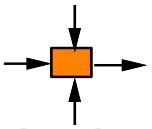
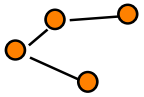
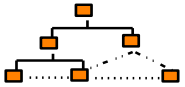
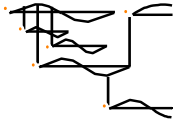
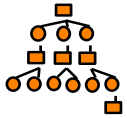
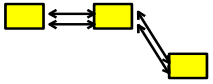
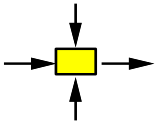
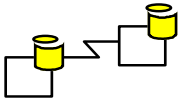
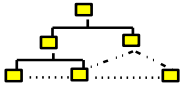
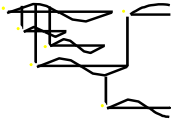
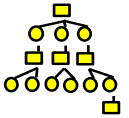
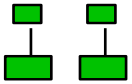
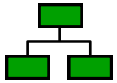
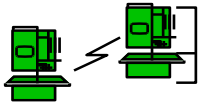
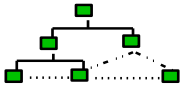
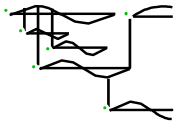
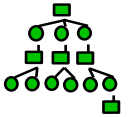
2011: The Zachman Framework for EA (v3)

Avancier

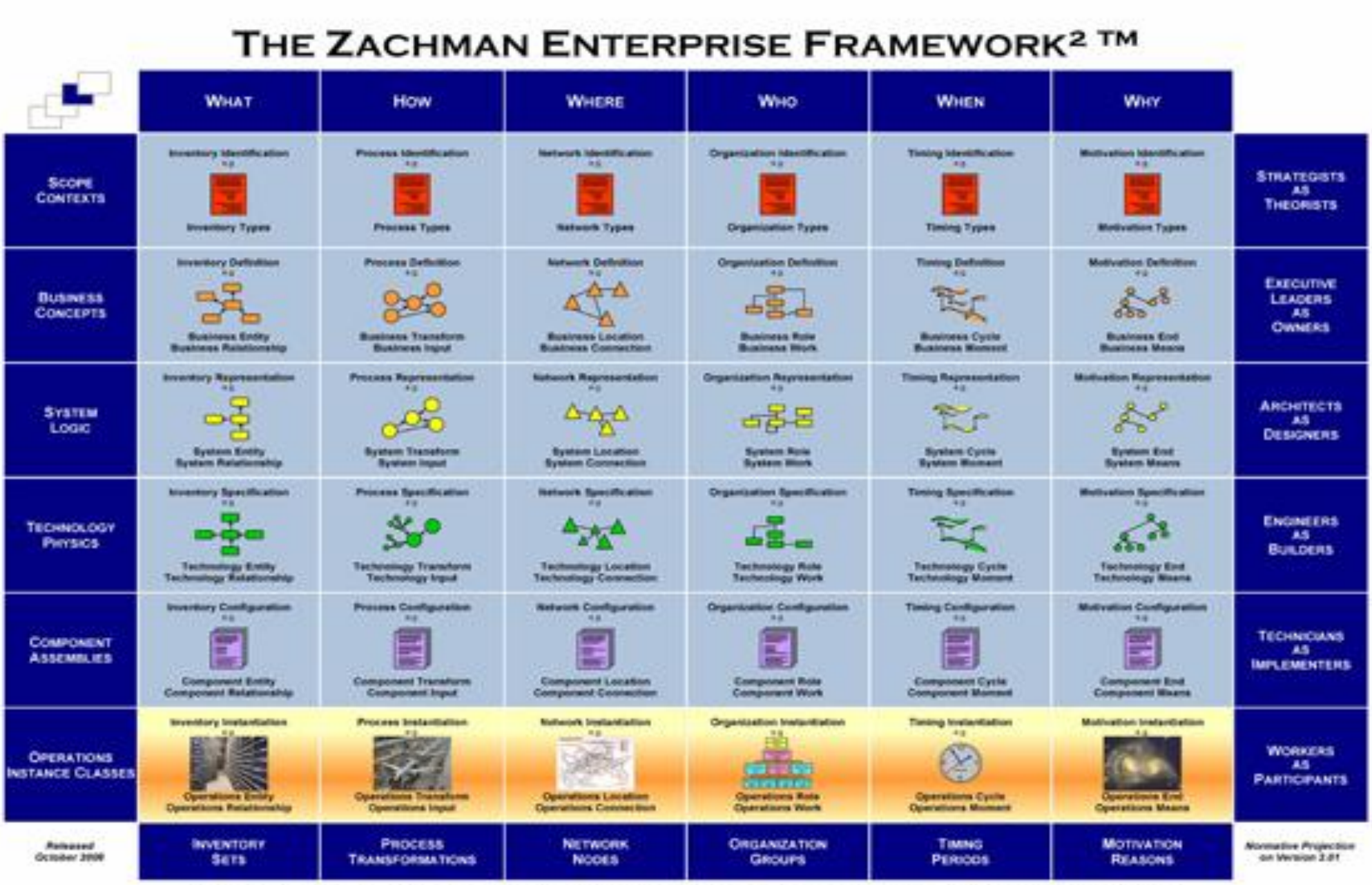
Zachman Framework v3		What (D)	How (P)	Where (L)	Who (O)	When	Why
Idealisation	Stakeholder perspective	Inventory sets	Process flows	Distribution networks	Responsibility assignments	Timing cycles	Motivation intentions
Scope Contexts	Executive	List inventory types	List process types	List distribution types	List responsibility types	List timing types	List motivation types
Business Concepts	Business management	Business entities & relationships	Business & input output	Business location & connection	Business role & work product	Business interval & moment	Business ends & means
System Logic	Architect	System entities & relationships	System & input output	System location & connection	System role & work product	System interval & moment	System ends & means
Technology Physics	Engineer	Technology entities & relationships	Technology input & output	Technology & location connection	Technology role & work product	Technology interval & moment	Technology ends & means
Tool components	Technician	Tool entities & relationships	Tool input & output	Tool location & connection	Tool role & work product	Tool interval & moment	Tool ends & means
Operations – Instance classes	Enterprise	Operations entities & relationships	Operations entities & relationships	Operations entities & relationships	Operations entities & relationships	Operations entities & relationships	Operations entities & relationships

Zachman Framework version 1 (1987)

ENTERPRISE ARCHITECTURE - A FRAMEWORK ™

	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*	
SCOPE (CONTEXTUAL) *Planner*	List of Things Important to the Business  ENTITY = Class of Business Thing	List of Processes the Business Performs  Function = Class of Business Process	List of Locations in which the Business Operates  Node = Major Business Location	List of Organizations Important to the Business  People = Major Organizations	List of Events Significant to the Business  Time = Major Business Event	List of Business Goals/Strat  Ends/Means=Major Bus. Goal/ Critical Success Factor	SCOPE (CONTEXTUAL) *Planner*
ENTERPRISE MODEL (CONCEPTUAL) *Owner*	e.g. Semantic Model  Ent = Business Entity Reln = Business Relationship	e.g. Business Process Model  Proc. = Business Process I/O = Business Resources	e.g. Logistics Network  Node = Business Location Link = Business Linkage	e.g. Work Flow Model  People = Organization Unit Work = Work Product	e.g. Master Schedule  Time = Business Event Cycle = Business Cycle	e.g. Business Plan  End = Business Objective Means = Business Strategy	ENTERPRISE MODEL (CONCEPTUAL) *Owner*
SYSTEM MODEL (LOGICAL) *Designer*	e.g. Logical Data Model  Ent = Data Entity Reln = Data Relationship	e.g. "Application Architecture"  Proc .= Application Function I/O = User Views	e.g. "Distributed System Architecture"  Node = I/S Function (Processor, Storage, etc) Link = Line Characteristics	e.g. Human Interface Architecture  People = Role Work = Deliverable	e.g. Processing Structure  Time = System Event Cycle = Processing Cycle	e.g., Business Rule Model  End = Structural Assertion Means =Action Assertion	SYSTEM MODEL (LOGICAL) *Designer*
TECHNOLOGY MODEL (PHYSICAL) *Builder*	e.g. Physical Data Model  Ent = Segment/Table/etc. Reln = Pointer/Key/etc.	e.g. "System Design"  Proc.= Computer Function I/O = Screen/Device Formats	e.g. "System Architecture"  Node = Hardware/System Software Link = Line Specifications	e.g. Presentation Architecture  People = User Work = Screen Format	e.g. Control Structure  Time = Execute Cycle = Component Cycle	e.g. Rule Design  End = Condition Means = Action	TECHNOLOGY CONSTRAINED MODEL (PHYSICAL) *Builder*
DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor*	e.g. Data Definition  Ent = Field Reln = Address	e.g. "Program"  Proc.= Language Stmt I/O = Control Block	e.g. "Network Architecture"  Node = Addresses Link = Protocols	e.g. Security Architecture  People = Identity Work = Job	e.g. Timing Definition  Time = Interrupt Cycle = Machine Cycle	e.g. Rule Specification  End = Sub-condition Means = Step	DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor*
FUNCTIONING ENTERPRISE	e.g. DATA	e.g. FUNCTION	e.g. NETWORK	e.g. ORGANIZATION	e.g. SCHEDULE	e.g. STRATEGY	FUNCTIONING ENTERPRISE

Zachman Framework version 2 (2009)



Zachman Framework version 3 (2011)

From IS to EA

- To model an information system is – necessarily – to model the business recorded in that information system
- So, it was easy for Zachman (in the mid 1990s) to relabel the framework as being for “Enterprise Architecture”

Not meant to be IS or IT-oriented

- "…the structure of the descriptive representations of buildings, airplanes and other complex industrial products."
- "Any appropriate approach, standard, role, method, technique, or tool may be placed in it.
- The schema can contain global plans as well as technical details, lists, and charts as well as natural language statements."

- Zachman expects completion of the cells to be determined by users of the framework.
- This freedom appeals to creative enterprise architects.

But in practice, EA *is* IS oriented

- "To keep the business from disintegrating, the concept of information systems architecture is becoming less of an option and more of a necessity.
- Enterprise Architecture provides the blueprint, or architecture, for the organization's information infrastructure."

- 1987 paper: proposed framework as a holder of information system descriptions.
- 1992 paper by Zachman and Sowa: says the framework had been adopted by systems analysts and database designers.
- Framework users still tend be information system-oriented
- *Because EA is about business processes that create and use business data*

Plotinus argued

- there is a process that works from perfect simplicity to complex imperfection.
- the complex derives from the simple

- “all of "creation" emanates from the one in succeeding stages of lesser and lesser perfection. These stages occur throughout time as a constant process.”
- “the multiple cannot exist without the simple. The "less perfect" must, of necessity, "emanate", or issue forth, from the "perfect" or "more perfect".

(Wikipedia)

But Zachman says: no sequence

- "the schema says nothing about the processes for developing viewpoints or conformant views, or the order in which they should be developed."

- The levels are not stages in a process or levels of top-down decomposition

- Note also that the abstraction from bottom to top is by idealisation, not by composition.

- Zachman has been known to say:
- “One day you [or your enterprise] will regret not having completed the schema”.

- By completed he means that every cell should contain architecture description,
- every level of architecture description should be completed, and
- every level should be completed to the lowest possible level of detail.

The “rules” of the Zachman Framework

Avancier

Rule 1:

Columns have no order

Rule 2:

Each column has a simple, basic model

Rule 3:

Basic model of each column is unique

Rule 4:

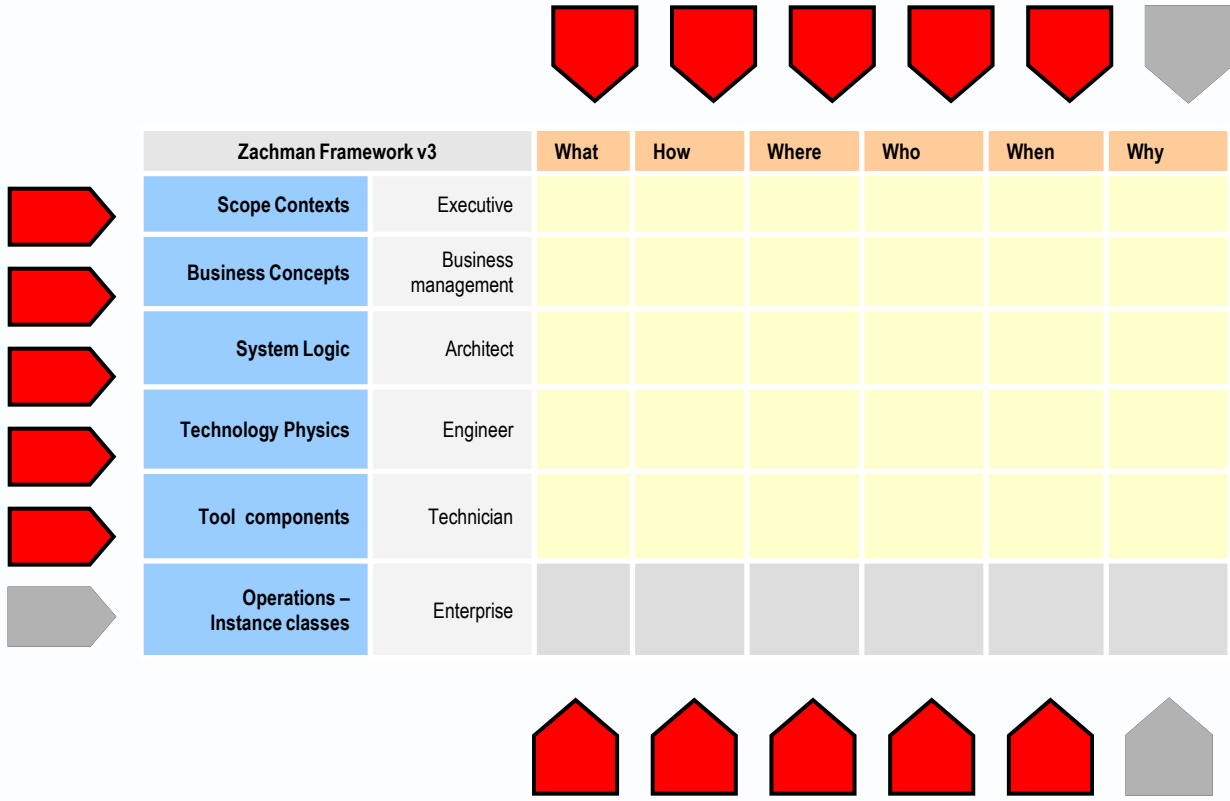
Each row represents a distinct view

Rule 5:

Each cell is unique

Rule 6:

Combining the cells in one row forms a complete description from that view



Zachman Framework v3		What	How	Where	Who	When	Why
Scope Contexts	Executive						
Business Concepts	Business management						
System Logic	Architect						
Technology Physics	Engineer						
Tool components	Technician						
Operations – Instance classes	Enterprise						

Bottom up more accurate than top down?

- Five levels of description is a lot of description
- At the bottom are tested working systems
- Higher level descriptions are flawed and approximate “soft systems”.
- The most accurate abstract descriptions are produced by reverse engineering from the bottom upwards.
- In practice, nobody can maintain perfect traceability between levels - unless by automated reverse engineering.

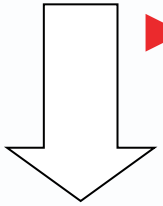
Simple cases are simple

- Given one facet of abstraction (idealisation)
- And abstraction in that direction only (not abstraction by composition)
- There could be1 to 1 mappings all the way up an down a column

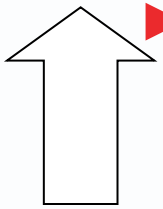
Zachman Framework v3		What	E.g.
Idealisation	**Stakeholder perspective**	**Inventory sets**	**E.g.**
Scope Contexts	Executive	List inventory types	
Business Concepts	Business management	Business entities & relationships	**'Employee' as conceptual entity type**
System Logic	Architect	System entities & relationships	**'Employee' as logical entity type**
Technology Physics	Engineer	Technology entities & relationships	**'Employee' as physical entity type**
Tool components	Technician	Tool entities & relationships	**'Employee' as database table name**
Operations - Instance classes	Enterprise	Operations entities & relationships	**Employee role played by John Smith**

- In the real world, 1 to 1 abstraction from real to ideal isn't practical
- There is and must be abstraction by composition and generalisation also

In practice: abstraction can work both down and up



- Downwards: a lower model contains additional details specific to a particular “physical” realisation of its next higher model.



- Upwards: a higher model may contain additional details that are not selected for realisation in the next lower model.

- So a downward refinement step may be only a “partial realisation”
  - It realises only part of a higher level model
  - And not all the way to the run-time system

In practice: a lower row might abstract from a higher one

- Can we fully realise a higher row in a lower row?
- That is, we study each excruciating detail of a higher row artefact and refine that detail (somehow) in one or more lower row artefacts?

- In practice, the highest level conceptual model may be only selectively realised in lower rows.

In practice

- The transformation of a description from one row to the next can be:
  - Multi-faceted – any or all of 5 or 6 different flavours of abstraction may be used at once.
  - Multi-directional – abstraction of one kind in one direction and refinement of the same or another kind in the opposite direction.
  - Many-to-many – there can be N-to-N cardinalities between types in adjacent layers.

- The result: a combinatorial explosion of the abstraction-refinement relations that can exist between artefacts in adjacent rows

In practice:

- Practitioners don’t distinguish abstraction types
- Their row to row transformations can be
  - multi-faceted,
  - multi-directional and
  - many-to-many
- And they don’t maintaining full traceability
- Perhaps that loose interpretation of the ZF is the best we can hope for?

How many possible 2D frameworks are possible?

- Make your own
- Perm any 2 of the 5 dimensions below.

Focus	Time	Abstraction by…		
Domain	**State**	**Composition**	**Idealisation**	**Generalisation**
Business	**Now**	**High level**	**Ideal**	**Generic**
Business	Baseline	Enterprise	Conceptual	Foundation
Data	Transition 1	Segment	Logical	Common System
Applications	Transition n	Solution	Physical	Industry
Technologies	Target	Detailed Design	Deployed Solutions	Organisation
Technology	**Future**	**Low level**	**Real**	**Specific**

Having said all that

- If you like the Zachman Framework, then
- you can with more or less difficulty populate the cells with artifacts mentioned in other EA frameworks

- Some ideas follow
- NONE OF WHAT FOLLOWS IS NECESSARILY IN ACCORD WITH WHAT ZACHMAN WOULD DO

Scopes of EA documentation in Zachman and TOGAF

Avancier

Zachman Framework	What	How	Where	Who	When	Why
	Inventory	Process	Network	Organisation	Time	Motivation
Scope Contexts Strategists & theorists						
Business Concepts Enterprise leaders & owners						
System Logic Architects & designers						
Technology Physics Engineers & builders						
Component assemblies Technicians & implementers						
Operations Instance classes Workers & Participants						

TOGAF-style EA repository
(abstract descriptions of human and computer activity systems;
descriptions of system structure and behaviour,
including component, process, interface and service types)

G

Implementation Governance

H

Architecture Change Management

Completed instructions and procedures
Coded HCI and business rules
Dev & test time libraries, schema and config.files

Run-time instances of human and technology components, processes and services

ITIL-style CMDB | **Asset Management** | **System & Network Monitoring** | **Business Databases** | **BPM**

The TOGAF artifacts might be roughly mapped to ZF

Phase A: Architecture Vision artifacts			Phase E Opportunities and Solutions
Stakeholder Map Matrix			Project Context Diagram
Value Chain Diagram			Benefits Diagram
Solution Concept Diagram			
Phase B Business Architecture artifacts	**Phase C Data Architecture artifacts**	**Phase C Application Architecture artifacts**	**Phase D Technology Architecture artifacts**
Organization/Actor Catalog	Data Entity/Data Component Catalog	Application Portfolio Catalog	Technical Reference Model
Driver/Goal/Objective Catalog		Interface Catalog	Technology Standards Catalog
Role Catalog			Technology Portfolio Catalog
Business Service/Function Catalog			
Location Catalog			
Process/Event/Control/Product Catalog			
Contract/Measure Catalog			
Business Interaction Matrix	Data Entity/Business Function Matrix	System/Organization Matrix	System/Technology Matrix
Actor/Role Matrix	System/Data Matrix	Role/System Matrix	
		System/Function Matrix	
		Application Interaction Matrix	
Business Footprint Diagram	Class Diagram	Application Communication Diagram	Environments and Locations Diagram
Business Service/Information Diagram	Data Dissemination Diagram	Application and User Location Diagram	Platform Decomposition Diagram
Functional Decomposition Diagram	Data Security Diagram (or matrix)	System Use-Case Diagram	Processing Diagram
Product Lifecycle Diagram	Data Migration Diagram	Enterprise Manageability Diagram	Networked Computing/Hardware Diagram
Goal/Objective/Service Diagram	Data Lifecycle Diagram	Process/System Realization Diagram	Communications Engineering Diagram
Business Use-Case Diagram	Class Hierarchy Diagram	Software Engineering Diagram	
Organization Decomposition Diagram		Application Migration Diagram	
Process Flow Diagram		Software Distribution Diagram	
Event Diagram			

A mapping of TOGAF artefacts to the Zachman Framework
(not including artefacts that obviously span more than one cell)

Avancier

	What	How	Where	Who	When	Why
	Inventory	Process	Network	Organisation	Time	Motivation
Scope Contexts Strategists & theorists	Business Service/Function Ctlg	Value Chain dgrm	Location Ctlg	Functional Decomposition dgrm	Event dgrm	Driver/Goal/Objective Ctlg Stakeholder Map Matrix
Business Concepts Enterprise leaders & owners	Business data model	Business Use-Case dgrm Process/Event/Control/Product Ctlg Process Flow dgrm	Business Interaction Matrix	Organization Decomposition dgrm Role Ctlg Organization/Actor Ctlg Actor/Role Matrix	Product Lifecycle dgrm	Goal/Objective/Service dgrm
System Logic Architects & designers	Application Portfolio Ctlg Interface Ctlg Data Entity/Data Component Ctlg	System Use-Case dgrm Process/System Realization dgrm	Application & User Location dgrm Application Interaction Matrix Application Communication dgrm		Application Migration dgrm Data Migration dgrm Data Lifecycle dgrm	Project Context dgrm Benefits dgrm
Technology Physics Engineers & builders	Technology Portfolio Ctlg		Networked Computing/Hardware dgrm Communications Engineering dgrm Environments & Locations dgrm			Technical Reference Model
Component assemblies Technicians & implementers	Class dgrm	Software Engineering dgrm	Software Distribution dgrm Processing dgrm Platform Decomposition dgrm			
Operations Instance classes Workers & participants						

Want to try it yourself? Fill out the ZF from the table

Avancier

Active network	Application availability	Application distribution & communication
Application use cases & services	Business data model	Business entities
Business events	Business goals & principles	Business locations
Business logistics	Business objectives & policies	Business org units
Business process flows	Business processes	Business requirements & rules
Business schedule	Data & time controls	Data in data stores
Database schema	Executing processes	Hardware nodes & platform apps
HCI	Identity & access controls	Implemented strategy
Logical data models	Network architecture	Operating schedule
Physical data models	Platform services	Program code
Roles & workflows	Rule design	Rule details & configuration
Running schedule	User devices & presentation layer	Working actors

	What	**How**	**Where**	**Who**	**When**	**Why**
	Inventory	Process	Network	Organisation	Time	Motivation
Scope Contexts Strategists & theorists						
Business Concepts Enterprise leaders & owners						
System Logic Architects & designers						
Technology Physics Engineers & builders						
Component assemblies Technicians & implementers						
Operations Instance classes Workers & participants						

A possible answer?

Avancier

	What	How	Where	Who	When	Why
	Inventory	Process	Network	Organisation	Time	Motivation
Scope Contexts Strategists & theorists	Business entities	Business functions & processes	Business locations	Business org units	Business events	Business goals & principles
Business Concepts Enterprise leaders & owners	Business data model	Business process flows	Business logistics	Roles & workflows	Business schedule	Business objectives & policies
System Logic Architects & designers	Logical data models	Application use cases & services	Application distribution & communication	HCI	Application availability	Business requirements & rules
Technology Physics Engineers & builders	Physical data models	Platform services	Hardware nodes & platform apps	User devices & presentation layer	Operating schedule	Rule design
Component assemblies Technicians & implementers	Database schema	Program code	Network architecture	Identity & access controls	Data & time controls	Rule details & configuration
Operations Instance classes Workers & participants	Data in data stores	Executing processes	Active network	Working actors	Running schedule	Implemented strategy

Plotinus may be discomforted to find that

- “the universe
- having started in a hugely complex big bang event – and
- being now complex enough to sustain information processing
- will probably end in a simple state called the big freeze.

- “A related scenario is heat death:
- the universe goes to a state of maximum entropy in which
- everything is evenly distributed, and
- there are no gradients —
- which are needed to sustain information processing,
- one form of which is life."
- (Wikipedia).