ZACHMAN INTERNATIONAL™

enterprise architecture

Engineering the Enterprise

THE ZACHMAN FRAMEWORK FOR ENTERPRISE ARCHITECTURE

Scope Contexts	Inventory Identification e.g. Inventory Types	Process Identification e.g. Process Types	Network Identification e.g. Network Types	Organization Identification e.g. Organization Types	Timing Identification e.g. Timing Types	Motivation Identification e.g. Motivation Types	Strategists as Theorists
Business Concepts	Inventory Definition e.g. Business Entity Business Relationship	Process Definition e.g. Business Transform Business Input	Network Definition e.g. Business Location Business Connection	Organization Definition e.g. Business Role Business Work	Timing Definition e.g. Business Cycle Business Moment	Motivation Definition e.g. Business End Business Means	Executive Leaders as Owners
System Logic	Inventory Representation e.g. System Entity System Relationship	Process Representation e.g. System Transform System Input	Network Representation e.g. System Location System Connection	Organization Representation e.g. System Role System Work	Timing Representation e.g. System Cycle System Moment	Motivation Representation e.g. System End System Means	Architects as Designers
Technology Physics	Inventory Specification e.g. Technology Entity Technology Relationship	Process Specification e.g. Technology Transform Technology Input	Network Specification e.g. Technology Location Technology Connection	Organization Specification e.g. Technology Role Technology Work	Timing Specification e.g. Technology Cycle Technology Moment	Motivation Specification e.g. Technology End Technology Means	Engineers as Builders
				Organization Configuration e.g.	Timing Configuration e.g. Component Cycle Component Moment	Motivation Configuration e.g. Component End Component Means	Technicians as Implementers

Enterprise Architecture

A Half-Day course by John A. Zachman

ZACHMAN CERTIFIED ENTERPRISE ARCHITECT

CERTIFIED EDUCATION

John A. Zachman
Biographical Sketch

John A. Zachman is the originator of the "Framework for Enterprise Architecture" which has received broad acceptance around the world as an integrative framework, or "periodic table" of descriptive representations for Enterprises. Mr. Zachman is not only known for this work on Enterprise Architecture, but is also known for his early contributions to IBM's Information Strategy methodology (Business Systems Planning) as well as to their Executive team planning techniques (Intensive Planning).

Mr. Zachman retired from IBM in 1990, having served them for 26 years. He is the Chief Executive Officer of his own education and consulting business, Zachman International. He also is Chairman of the Board of Zachman Framework Associates, a worldwide consortium managing conformance to The Zachman Framework™ principles and Chief Executive Officer of the Zachman Institute for Framework Advancement (ZIFA), an organization dedicated to advancing the conceptual and implementation states of the art in Enterprise Architecture.

Mr. Zachman serves on the Executive Council for Information Management and Technology (ECIMT) of the United States Government Accountability Office (GAO). He is a Fellow for the College of Business Administration of the University of North Texas. He serves on the Advisory Board for the Data Resource Management Program at the University of Washington and on the Advisory Board of the Data Administration Management Association International (DAMA-I) from whom he was awarded the 2002 Lifetime Achievement Award. He was awarded the 2004 Oakland University, Applied Technology in Business (ATIB), Award for IS Excellence and Innovation.

Mr. Zachman has been focusing on Enterprise Architecture since 1970 and has written extensively on the subject. He is the author of the book, "The Zachman Framework for Enterprise Architecture™: A Primer on Enterprise Engineering and Manufacturing." He has facilitated innumerable executive team planning sessions. He travels nationally and internationally, teaching and consulting, and is a popular conference speaker, known for his motivating messages on Enterprise Architecture issues. He has spoken to many thousands of enterprise managers and information professionals on every continent.

In addition to his professional activities, Mr. Zachman serves on the Elder Council of the Church on the Way (First Foursquare Church of Van Nuys, California), the Board of Directors of Living Way Ministries, a radio and television ministry of the Church on the Way, the President's Cabinet of the King's College and Seminary, the Board of Directors of the Los Angeles Citywide Children's Christian Choir and on the Board of Directors of Native Hope International, a Los Angeles-based ministry to the Native American people.

Prior to joining IBM, Mr. Zachman served as a line officer in the United States Navy and is a retired Commander in the U. S. Naval Reserve. He chaired a panel on "Planning, Development and Maintenance Tools and Methods Integration" for the U.S. National Institute of Standards and Technology. He holds a degree in Chemistry from Northwestern University, has taught at Tufts University, has served on the Board of Councilors for the School of Library and Information Management at the University of Southern California, as a Special Advisor to the School of Library and Information Management at Emporia State University, and on the Advisory Council to the School of Library and Information Management at Dominican University.

Enterprise Architecture

Enterprise Design Objectives: Complexity and Change

John A. Zachman
Zachman International
2222 Foothill Blvd. Suite 337
La Canada, Ca. 91011
818-244-3763

Introduction

Enterprise Architecture presently appears to be a grossly misunderstood concept among management.

It is NOT an Information Technology issue.
It is an ENTERPRISE issue.

It is likely perceived to be an Information Technology issue as opposed to a Management issue for two reasons:

A. Awareness of it tends to surface in the Enterprise through the Information Systems community.

B. Information Technology people seem to have the skills to do Enterprise Architecture if any Enterprise Architecture is being or is to be done.

Origins of Ent. Arch.

Frederick Taylor "Principles of Scientific Management" 1911

Walter A. Shewhart "The Economic Control of Quality of Manufactured Product" 1931 (Dr. Edward Demming's Mgr.)

Peter Drucker "The Practice of Management" 1954

Jay Forrester "Industrial Dynamics" 1961

Peter Senge "The Fifth Discipline" 1990

Eric Helfert "Techniques of Financial Analysis" 1962

Robert Anthony "Planning and Control Systems: A Framework for Analysis" 1965

Sherman Blumenthal "Management Information Systems: A Framework for Planning and Development" 1969

Alvin Toffler "Future Shock" 1970

George Steiner "Comprehensive Managerial Planning" 1972

Etc., etc., etc.

"Enterprise"

There are two implications to the word "Enterprise":

I. Scope

The broadest possible boundary of the Enterprise.
The Enterprise in its entirety.
Enterprise-wide in scope.
The whole thing.

II. Content

ENTERPRISE Architecture is for ENTERPRISES.
Enterprise Architecture has nothing to do with the Enterprise's systems or its information technology (except as they may constitute Row 4 constraints).
The end object is to engineer and manufacture the ENTERPRISE, NOT simply to build and run systems.

"ENTERPRISE" ACTUALLY MEANS "ENTERPRISE"

"Architecture"

Architecture ... what is it?

Some people think this is Architecture:

That is a common

MISCONCEPTION

(Note: This same misconception about Enterprises is what leads people to misconstrue Enterprise Architecture as being big, monolithic, static, inflexible and unachievable and ... it takes too long and costs too much.)

"Architecture"

This is the RESULT of architecture.
In the RESULT you can see the Architect's "architecture".
The RESULT is an implementation, an instance.

"Architecture" IS the set of descriptive representations relevant for describing a complex object (actually, any object) such that an instance of the object can be created and such that the descriptive representations serve as the baseline for changing an object instance (assuming that the descriptive representations are maintained consistent with the instantiation).

"Architecture"

If the object you are trying to create is simple, you can see the whole thing all at one time, and it is not likely to change, (e.g. a log cabin, a program, etc.), then you don't need Architecture. All you need is a tool (e.g. an ax, a compiler, etc.), some raw material (e.g. a forest, some data, etc.) and some time (then, build log cabins, write programs, etc.).

On the other hand, if the object is complex, you can't see it in its entirety at one time and it is likely to change considerably over time (e.g. a hundred story building, an Enterprise, etc.), now you need Architecture.

In short, the reasons you need Architecture:

COMPLEXITY AND CHANGE

"Architecture"

COMPLEXITY

If you can't describe it, you can't create it (whatever "it" is).

CHANGE

If you don't retain the descriptive representations after you create them (or if you never created them in the first place) and you need to change the resultant implementation, you have only three options:

A. Change the instance and see what happens. (High risk!)
B. Recreate ("reverse engineer") the architectural representations from the existing ("as is") implementation. (Takes time and costs money!)
C. Scrap the whole thing and start over again.

"Architecture"

There is not a single descriptive representation for a complex object ... there is a SET of descriptive representations.

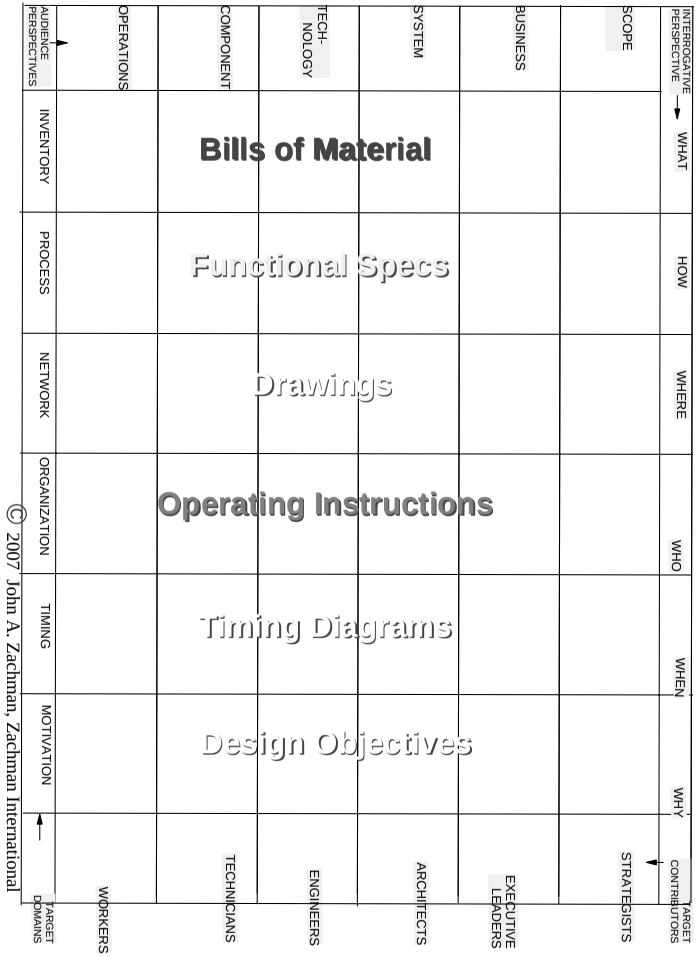
Descriptive representations (of anything) typically include "Abstractions":

A. Bills of Material (What)
B. Functional Specs (How)
C. Drawings (Where)
D. Operating Instructions (Who)
E. Timing Diagrams (When)
F. Design Objectives (Why)

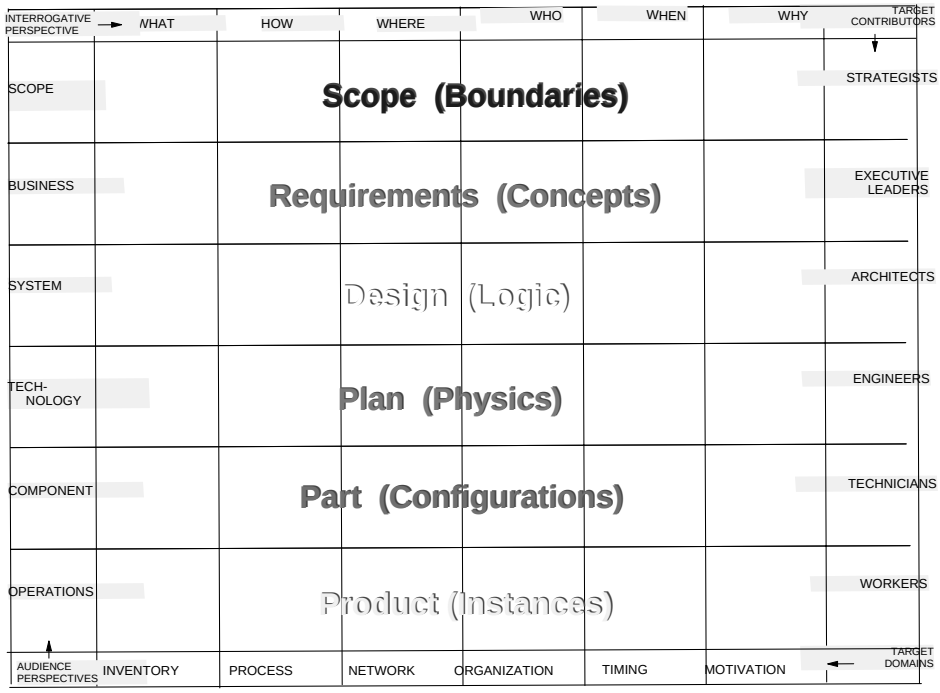
as well as Perspectives:

1. Scoping Boundaries (Strategists)
2. Requirement Concepts (Owners)
3. Design Logic (Designers)
4. Plan Physics (Builders)
5. Part Configurations (Implementers)
   and the
6. Product Instances (Operators)

Abstractions



Perspectives



"Architecture"

There is not a single descriptive representation for a complex object ... there is a SET of descriptive representations.

Descriptive representations (of anything) typically include "Abstractions":

A. Bills of Material (What)
B. Functional Specs (How)
C. Drawings (Where)
D. Operating Instructions (Who)
E. Timing Diagrams (When)
F. Design Objectives (Why)

as well as Perspectives:

1. Scope Boundaries (Strategists)
2. Requirement Concepts (Owners)
3. Design Logic (Designers)
4. Plan Physics (Builders)
5. Part Configurations (Implementers) and the
6. Product Instances (Operators)

"Architecture" In General

"**Architecture**" (for anything) would be the total set of descriptive representations (models) relevant for describing a complex object such that it can be created and that constitute a baseline for changing the object after it has been instantiated. The relevant descriptive representations would necessarily have to include all the intersections between:

the "Abstractions":

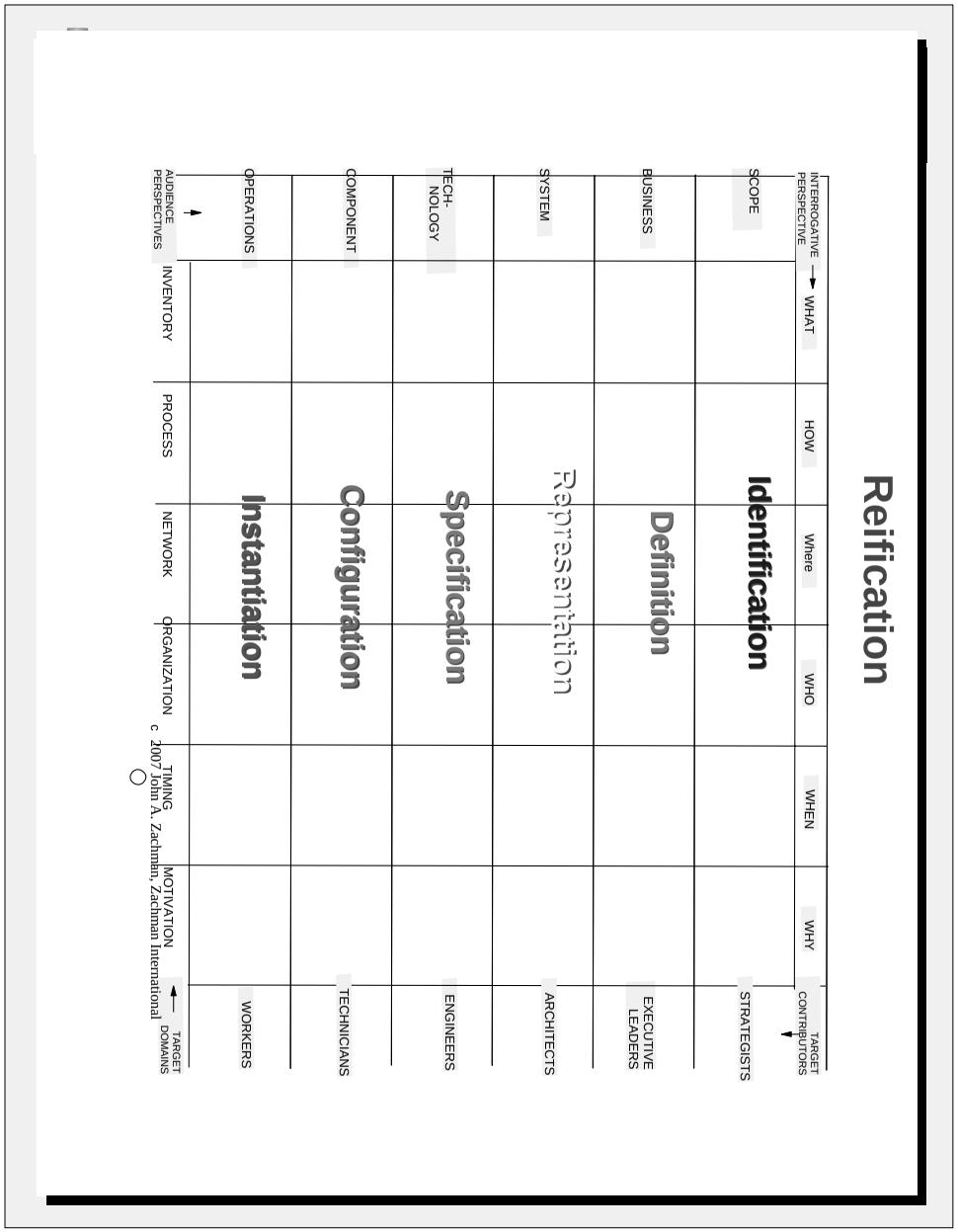
A. Bills of Material (What)
B. Functional Specs (How)
C. Drawings (Where)
D. Operating Instructions (Who)
E. Timing Diagrams (When)
F. Design Objectives (Why)

and the Perspectives:

1. Scoping Boundaries (Identification)
2. Requirement Concepts (Definition)
3. Design Logic Representation)
4. Plan Physics Specification)
5. Part Configurations (Configuration)

resulting in the

6. Product Instances (Instantiation)



"Enterprise Architecture"

Therefore **"Enterprise Architecture"** would be the total set of descriptive representations (models) relevant for describing an Enterprise, that is, the descriptive representations required to create (a coherent, optimal) Enterprise and required to serve as a baseline for changing the Enterprise once it is created. The total set of relevant descriptive representations would necessarily have to include all the intersections between the

Abstractions:

A. Inventory Models (Bills of Material)
B. Process Models (Functional Specs)
C. Geographic Models (Drawings)
D. Work Flow Models (Operating Instructions)
E. Cyclical Models (Timing Diagrams)
F. Objective Models (Design Objectives)

and the Perspectives:

1. Scope Boundaries (Scoping Boundaries)
2. Business Models (Requirement Concepts)
3. System Models (Design Logic)
4. Technology Models (Plan Physics)
5. Tooling Configurations (Part Configurations)

resulting in the

6. The Enterprise Implementation (Product Instance)

"Enterprise Architecture"

The total set would necessarily have to include Abstractions:

WHAT

Inventory Models equal Bills of Materials
(Entity Models
and Data Models ARE Bills of Material)

HOW

Process Models equal Functional Specs
(Transformation Models)

WHERE

Network Models equal Drawings
(Geographic Models) (Geometry)
(Distribution Models)

WHO

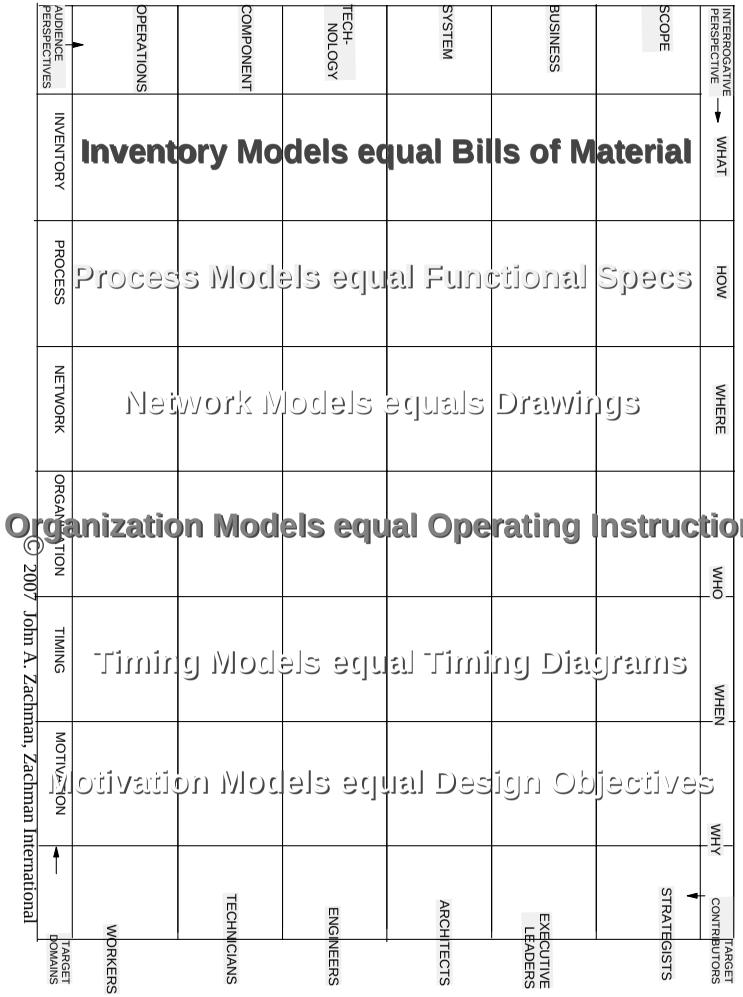
Organization Models equal Operating Instructions
(Work Flow Models)
(Presentation Architecture)

WHEN

Timing Models equal Timing Diagrams
(Control Structures)
(Cyclical Models)
(Dynamics Models)

WHY

Motivation Models equal Design Objectives



"Enterprise Architecture"

The total set would necessarily have to include Perspectives:

STRATEGISTS

Scope Boundaries equal Scope Boundaries

("CONOPS" or Concepts Package)

EXECUTIVE LEADERS

Business Models equal Requirement Concepts

(Concepts Models) (Customer's Usage) ("Computation Independent")

ARCHITECTS

System Models equal Design Logic

(Logic Models) (Engineering Descriptions) ("Platform Independent")

ENGINEERS

Technology Models equal Plan Physics

(Physics Models) (Mfg. Eng. Descriptions) ("Platform Specific")

TECHNICIANS

Tooling Configurations equal Part Configurations

(Vendor Product Specific) (Machine Tool Specific)

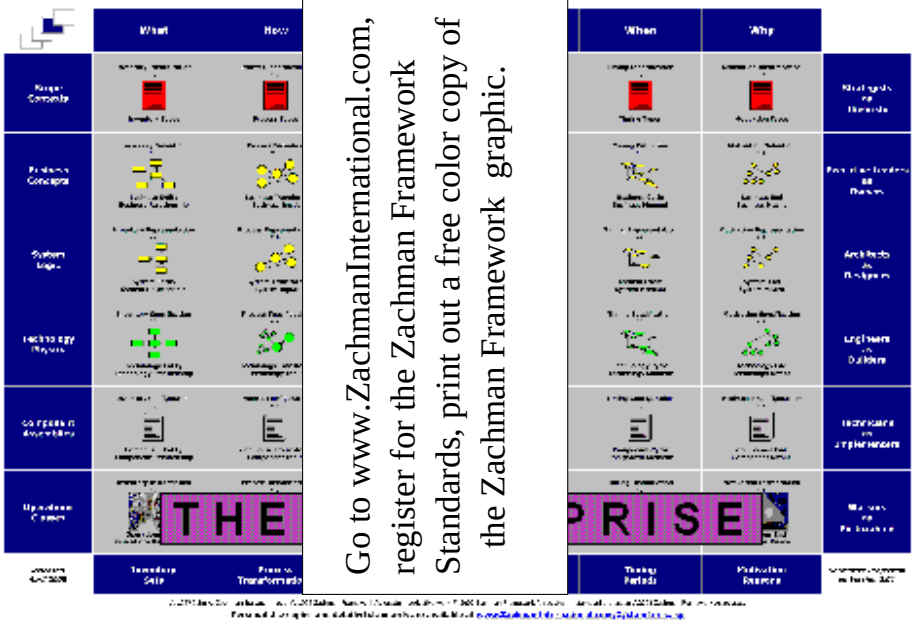
WORKERS

Enterprise Implementation equals Product Instance

(Operations Instances)

ENTERPRISE ARCHITECTURE

THE ZACHMAN ENTERPRISE FRAMEWORK² ™



Go to www.ZachmanInternational.com, register for the Zachman Framework Standards, print out a free color copy of the Zachman Framework graphic.

Perspectives

INTERROGATIVE PERSPECTIVE →	WHAT	HOW	WHERE	WHO	WHEN	WHY	TARGET CONTRIBUTORS ↓
SCOPE	**Scope Boundaries equals Scope Boundaries**						STRATEGISTS
BUSINESS	**Business Models equal Requirement Concepts**						EXECUTIVE LEADERS
SYSTEM	Systems Models equal Design Logic						ARCHITECTS
TECH-NOLOGY	**Technology Models equal Plan Physics**						ENGINEERS
COMPONENT	**Tooling Configurations equal Part Configurations**						TECHNICIANS
OPERATIONS	Enterprise Implementation equals Product Instances						WORKERS
AUDIENCE PERSPECTIVES ↑	INVENTORY	PROCESS	NETWORK	ORGANIZATION	TIMING	MOTIVATION	← TARGET DOMAINS

Architecture Is Architecture

I learned about architecture for Enterprises by looking at architecture for:

Airplanes, Buildings, Locomotives, Computers,
... Complex Industrial Products

It is all the same ...

Bills of Material, Functional Specs, Drawings, ... etc.
Requirements, Schematics, Blueprints, ... etc.

ENTERPRISES have:

Bills of Material, Functional Specs, Drawings, ... etc.

ENTERPRISES have:

Requirements, Schematics, Blueprints, ... etc.

The Engineering Design Artifacts (the descriptive representations of anything) fall into a two dimensional classification system:

A. The focus of the description (Abstraction)
(What, How, Where, Who, When, Why)

B. The usage of the description (Perspective)
(Owner, Designer, Builder)

Architecture Is Architecture

I simply put Enterprise names on the same descriptive representations relevant for describing anything.

Why would anyone think
that the descriptions of an Enterprise
are going to be any different
from the descriptions of anything else
humanity has ever described?

ARCHITECTURE
IS ARCHITECTURE
IS ARCHITECTURE

I don't think Enterprise Architecture is arbitrary ...
and it is not negotiable.

My opinion is, we ought to accept the definitions of Architecture that the older disciplines of Architecture and Construction, Engineering and Manufacturing have established and focus our energy on learning how to use them to actually engineer Enterprises.

Ontology

The Zachman Framework schema technically is an ontology - a theory of the existence of a structured set of essential components of an object for which explicit expression is necessary (is mandatory?) for designing, operating and changing the object (the object being an Enterprise, a department, a value chain, a "sliver," a solution, a project, an airplane, a building, a bathtub or whatever or whatever).

The Zachman Framework is NOT a methodology for creating the implementation (an instantiation) of the object (i.e. the Framework is an ontology, not a methodology).

A Framework is a STRUCTURE.
(A Structure DEFINES something.)
An Ontology is a theory of existance - what IS
An Ontology IS a Structure.

A Methodology is a PROCESS.
(A Process TRANSFORMS something.)

A Structure IS NOT A Process
A Process IS NOT a Structure.

Process

A Process TRANSFORMS something.

This is a Process:

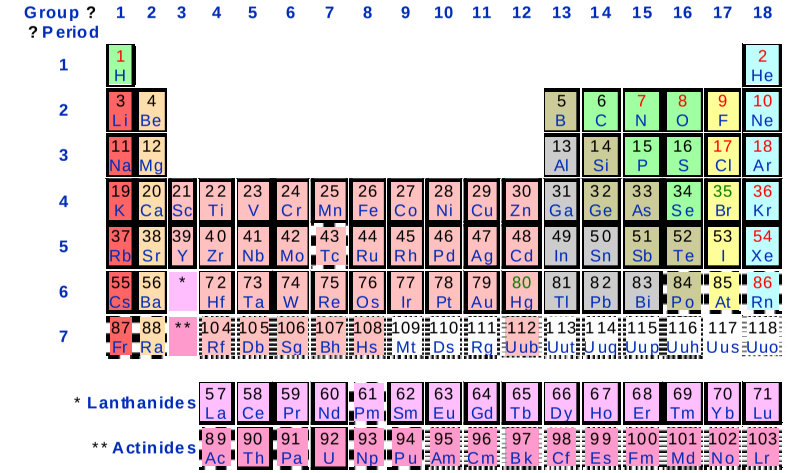
Add Bleach to an Alkali and
it is transformed into Saltwater.

Ontology

A Structure DEFINES something.

This is a Structure:

Standard periodic table



This is a Structure, an ontological structure ... a fixed, structured set of elemental components that exist of which any and every compound must be composed.

The Periodic Table provides precise DEFINITION.

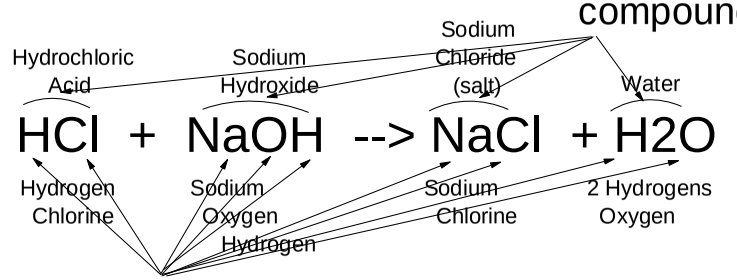
Chemistry - A Science

A Process TRANSFORMS something.

This is a PROCESS:

(Compounds are virtually infinite)

These are compounds



These are elements

(Elements come from the Ontology - finite)

A Process based on an ONTOLOGICAL structure will be repeatable and predictable - A SCIENCE.

Process

A Process TRANSFORMS something.

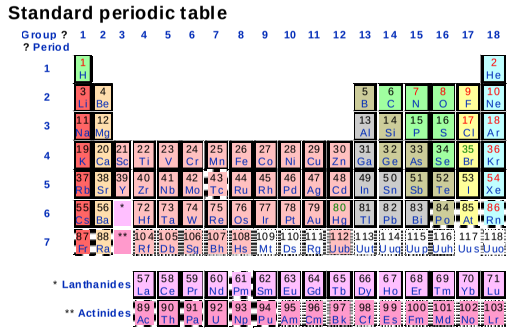
This is a Process:

Add Bleach to an Alkali and
it is transformed into Saltwater.

A Process with no ontological structure is ad hoc, fixed and dependent on practitioner skills. This is NOT a science. **It is ALCHEMY**.

Ontology vs Process

An Ontology



IS NOT

Hydrochloric Acid		Sodium Hydroxide		Sodium Chloride (salt)		Water
HCl	+	NaOH	-->	NaCl	+	H2O
Hydrogen Chlorine		Sodium Oxygen Hydrogen		Sodium Chlorine		2 Hydrogens Oxygen

A Process

and a Process IS NOT an Ontology.

Ontology vs Methodology

The Framework does not imply anything about:

a. whether you do Architecture or whether you simply build systems (that is, whether you build Primitive Models, the single variable intersections between the Abstractions and the Perspectives or whether you build multi-variable, composite models made up of components of several Primitive Models)

b. how you do Architecture (top-down, bottom-up, left to right, right to left, where to start, etc., etc.)

c. the long term/short term trade-off relative to instantiating the expression of the components of the object (i.e. what is formalized in the short term for implementation purposes versus what is engineered for long term reuse).

d. how much flexibility you want for producing composite models (Enterprise implementations) from your Enterprise Architecture (primitive models), that is, how constrained (little flexibility) or unconstrained (much flexibility) you make the horizontal, integrative relationships between the Cell components across the Rows and the vertical, transformational relationships of the Cell components down the Columns.

(These are significant, identifyable methodological choices ... not prescriptions of the Framework.)

The Framework Is a Schema

The Fmwrk is a two-dimensional classification system for ENTERPRISE descriptive representations NOT I/S.

The classification scheme for each axis grew up quite independently from the Framework application.

The classification for each axis is:

a. Comprehensive
b. Non-redundant

Therefore, each cell of the Framework is:

a. Unique
b. "Primitive" (one single Abstraction by one single Perspective)

and the total set of cells is complete.

The Framework logic is universal, independent of its application - totally neutral relative to methods/tools.

The Framework is a "normalized" schema ...
... NOT a matrix.

That's what makes it a good analytical tool.

Introducing a Metaphor

A reasonable metaphor for the Framework is the Periodic Table. The Periodic Table is an ontology ... a schema ... a normalized schema ... one element goes in one and only one cell. The Periodic Table doesn't do anything. It reflects nature. The Periodic Table (an ontology) is used by Chemists (practitioners) to define a Process (a methodology) for producing compounds (results, implementations, composites). If an alchemist uses the Periodic Table to define the process, the process can be dynamically defined (or re-defined) and will be repeatable and produce predictable results ... and the alchemist will become a Chemist. On the other hand, if the alchemist ignores the Periodic Table, they can define a process (a methodology) that will produce results, point-in-time solutions, based on their own skills and experience. The process (methodology) will be fixed (not changeable) and the alchemist will forever remain an alchemist.

Practitioners (methodologists) are constrained
by time and results.
Theoreticians (scientists) are constrained
by natural laws and integrity.

The Periodic Table Metaphor

Before Mendeleev figured out the Periodic table, Alchemists (practitioners) could create compounds based on their experience ... whatever worked. After Mendeleev figured out the Periodic Table, Chemistry became a science. Creating compounds became predictable and repeatable based on the natural laws (Physics) expressed in the Periodic Table. Within 50 years, the Chemists and Physicists (practitioners) were splitting atoms.

If I am right that Architecture is Architecture is Architecture, and if my work understanding the underlying primitives (elements) of Architecture correctly reflects the natural laws of classification and has integrity, maybe my Framework will form the basis for making Enterprise Architecture a science ... and maybe in 50 years, the methodologists (practitioners) will be able to engineer Enterprises to be assembled to order from reusable "primitive" components dynamically. I don't know. I hope so. We'll probably know in 50 years.

ZachmanInternational.com

I have provided Enterprise Architecture resources to help you with your Enterprise Architecture endeavors including:

a. Zachman Enterprise Framework Standards (detailed contents for the Enterprise Framework Cells).
b. Printable A4 version (8 1/2 X 11) of the Enterprise Framework graphic.
c. Several topical articles I have written including, "Why Framework Standards", "What is Enterprise Architecture", "My Definition of the Zachman Framework", etc.
d. Calendar for my public appearances and seminars.
e. Information about my electronic book, "The Zachman Framework: A Primer for Enterprise Engineering and Manufacturing".
f. A biography for John A. Zachman
g. Links to other Zachman International activities.
h. Zachman Certification Program
i. Etc., etc.

The only website that contains Zachman-related material created by or specifically approved by me is:

www.ZachmanInternational.com

Enterprise Architecture

Conclusions

1965 Systems Problems

1. Didn't meet Requirements. (not "aligned")
2. The data was no good:
   - Not consistent from system to system.
   - Not accurate.
   - Not accessible.
   - Too late.
3. Couldn't change the system. (Inflexible)
4. Couldn't change the technology. (Not adaptable)
5. Couldn't change the business. (Couldn't change the system or the technology so couldn't change business.)
6. Little new development (80% $ for maintenance)
7. Took too long.
8. Cost too much.
9. Always over budget.
10. Always missed schedules.
11. DP budget out of control.
12. Too complicated - can't understand it, can't manage it.
13. Just frustrating.

(Adapted from Doug Erickson)

2009 Systems Problems

1. Don't meet Requirements. (not "aligned")
2. The data is no good:
   - Not consistent from system to system.
   - Not accurate.
   - Not accessible.
   - Too late.
3. Can't change the system. (Inflexible)
4. Can't change the technology. (Not adaptable)
5. Can't change the business. (Can't change the system or the technology so can't change business.)
6. Little new development (80% $ for maintenance)
7. Takes too long.
8. Costs too much.
9. Always over budget.
10. Always missed schedules.
11. IT budget out of control.
12. Too complicated - can't understand it, can't manage it.
13. Just frustrating.

(*Adapted from Doug Erickson*)

It's Funny ...

COBOL didn't fix those problems!

MVS didn't fix those problems!

Virtual Memory didn't fix those problems!

IMS, DB2, Oracle, Sybase, Access, Fortran, PL/1, ADA, C++, Visual Basic, JAVA 2, 360's, 390's, MPP's, DEC VAX's, H200's, Crays, PC's, MAC's, Distributed Processing, didn't fix those problems!

Word, Excel, Powerpoint, Outlook Express, eMAIL, DOS, Windows 95, 98, 2000, NT, ME, XP, Unix, Linux, Object Oriented, COM, DCOM, CORBA, EDI, HTML, XML, UML, the Internet, B2B, B2C, Portals, Browsers didn't fix those problems!

IEF, IEW, ADW, ERWIN, POPKIN, Rational, PTECH, Rochade, Platinum, Design Bank, Data Warehouse, SAP, Baan, Peoplesoft, Oracle Financials, BSP, ISP, EAP, EAI didn't fix those problems!

And, I doubt that Web Services, .Net, Websphere, Extreme Programming, Service Oriented Architecture or Component Development (whatever that is) is going to fix the problems.

IT MAKES ONE WONDER IF THERE ACTUALLY IS A TECHNICAL SOLUTION TO THE PROBLEM!!!

Engineering Problem

I'm not saying that there is anything wrong with any of these technologies.

In fact, any or all of them may well be very good ...

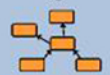
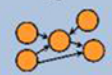
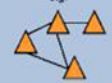
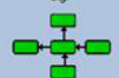
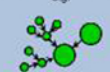
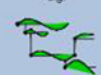
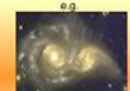
In fact, you may not be able to solve the Enterprise problem without employing some of these technologies.

However,
The Enterprise problem is an ENGINEERING problem,
NOT a technical problem.

My perception is that it is going to take actual work, ENGINEERING work, to solve the problem. My plan would be to start building out models, PRIMITIVE models, engineering them for alignment, integration, flexibility, reduced time-to-market, etc., etc., etc.

What would be YOUR plan for solving the problems???

THE ZACHMAN ENTERPRISE FRAMEWORK² ™

	WHAT	HOW	WHERE	WHO	WHEN	WHY	
SCOPE	Inventory Identification e.g.  Inventory Types	Process Identification e.g.  Process Types	Network Identification e.g.  Network Types	Organization Identification e.g.  Organization Types	Timing Identification e.g.  Timing Types	Motivation Identification e.g.  Motivation Types	STRATEGISTS
BUSINESS	Inventory Definition e.g.  Business Entity Business Relationship	Process Definition e.g.  Business Transform Business Input	Network Definition e.g.  Business Location Business Connection	Organization Definition e.g.  Business Role Business Work	Timing Definition e.g.  Business Cycle Business Moment	Motivation Definition e.g.  Business End Business Means	EXECUTIVE LEADERS
SYSTEM	Inventory Representation e.g.  System Entity System Relationship	Process Representation e.g.  System Transform System Input	Network Representation e.g.  System Location System Connection	Organization Representation e.g.  System Role System Work	Timing Representation e.g.  System Cycle System Moment	Motivation Representation e.g.  System End System Means	ARCHITECTS
TECHNOLOGY	Inventory Specification e.g.  Technology Entity Technology Relationship	Process Specification e.g.  Technology Transform Technology Input	Network Specification e.g.  Technology Location Technology Connection	Organization Specification e.g.  Technology Role Technology Work	Timing Specification e.g.  Technology Cycle Technology Moment	Motivation Specification e.g.  Technology End Technology Means	ENGINEERS
COMPONENT	Inventory Configuration e.g.  Component Entity Component Relationship	Process Configuration e.g.  Component Transform Component Input	Network Configuration e.g.  Component Location Component Connection	Organization Configuration e.g.  Component Role Component Work	Timing Configuration e.g.  Component Cycle Component Moment	Motivation Configuration e.g.  Component End Component Means	TECHNICIANS
OPERATIONS	Inventory Instantiation e.g.  Operations Entity Operations Relationship	Process Instantiation e.g.  Operations Transform Operations Input	Network Instantiation e.g.  Operations Location Operations Connection	Organization Instantiation e.g.  Operations Role Operations Work	Timing Instantiation e.g.  Operations Cycle Operations Moment	Motivation Instantiation e.g.  Operations End Operations Means	WORKERS
Released October 2007	INVENTORY	PROCESS	NETWORK	ORGANIZATION	TIMING	MOTIVATION	Version 2.01