Enterprise Architecture

# The Zachman Framework: Solving General Management Problems

John A. Zachman
Zachman International

# Agenda

I. The Zachman Framework
II. A Zachman Framework Story
III. An Illustration of Primitive Modeling Concepts
IV. Methodology for Solving General Management Problems
V. Conclusions

# The Information Age

"The next information revolution is well underway. But it is not happening where information scientists, information executives, and the information industry in general are looking for it. It is not a revolution in technology, machinery, techniques, software, or speed. It is a revolution in CONCEPTS."

Peter Drucker. Forbes ASAP, August 24, 1998

Introduction to
Enterprise Architecture

# The Framework for Enterprise Architecture

John A. Zachman
Zachman International

# The Zachman Framework for Enterprise Architecture™

## *The Enterprise Ontology* ™

*Version 3.0*

| Classification Names / Audience Perspectives | What | How | Where | Who | When | Why | Classification Names / Model Names |
|---|---|---|---|---|---|---|---|
| | *Composite Integrations* → | | ← *Alignment* → | | ← *Composite Integrations* | | |
| **Executive Perspective** *(Business Context Planners)* | **Inventory Identification** e.g. *List: Inventory Types* | **Process Identification** e.g. *List: Process Types* | **Distribution Identification** e.g. *List: Distribution Types* | **Responsibility Identification** e.g. *List: Responsibility Types* | **Timing Identification** e.g. *List: Timing Types* | **Motivation Identification** e.g. *List: Motivation Types* | **Scope Contexts** *(Scope Identification Lists)* |
| *Alignment Transformations* | | | | | | | *Alignment Transformations* |
| **Business Mgmt Perspective** *(Business Concept Owners)* | **Inventory Definition** e.g. *Business Entity / Business Relationship* | **Process Definition** e.g. *Business Transform / Business Input/Output* | **Distribution Definition** e.g. *Business Location / Business Connection* | **Responsibility Definition** e.g. *Business Role / Business Work Product* | **Timing Definition** e.g. *Business Interval / Business Moment* | **Motivation Definition** e.g. *Business End / Business Means* | **Business Concepts** *(Business Definition Models)* |
| | **Inventory Representation** e.g. *System Entity / System Relationship* | **Process Representation** e.g. *System Transform / System Input /Output* | **Distribution Representation** e.g. *System Location / System Connection* | **Responsibility Representation** e.g. *System Role / System Work Product* | **Timing Representation** e.g. *System Interval / System Moment* | **Motivation Representation** e.g. *System End / System Means* | **System Logic** *(System Representation Models)* |
| **Engineer Perspective** *(Business Physics Builders)* | **Inventory Specification** e.g. *Technology Entity / Technology Relationship* | **Process Specification** e.g. *Technology Transform / Technology Input /Output* | **Distribution Specification** e.g. *Technology Location / Technology Connection* | **Responsibility Specification** e.g. *Technology Role / Technology Work Product* | **Timing Specification** e.g. *Technology Interval / Technology Moment* | **Motivation Specification** e.g. *Technology End / Technology Means* | **Technology Physics** *(Technology Specification Models)* |
| **Technician Perspective** *(Business Component Implementers)* | **Inventory Configuration** e.g. *Tool Entity / Tool Relationship* | **Process Configuration** e.g. *Tool Transform / Tool Input /Output* | **Distribution Configuration** e.g. *Tool Location / Tool Connection* | **Responsibility Configuration** e.g. *Tool Role / Tool Work Product* | **Timing Configuration** e.g. *Tool Interval / Tool Moment* | **Motivation Configuration** e.g. *Tool End / Tool Means* | **Tool Components** *(Tool Configuration Models)* |
| **Enterprise Perspective** *(Users)* **The Enterprise** | **Inventory Instantiations** *Operations Entities / Operations Relationships* | **Process Instantiations** *Operations Transforms / Operations In/Outputs* | **Distribution Instantiations** *Operations Locations / Operations Connections* | **Responsibility Instantiations** *Operations Roles / Operations Work Products* | **Timing Instantiations** *Operations Intervals / Operations Moments* | **Motivation Instantiations** *Operations Ends / Operations Means* | **Operations Instances** *(Implementations)* **The Enterprise** |
| | *Composite Integrations* → | | ← *Alignment* → | | ← *Composite Integrations* | | |
| *Audience Perspectives / Enterprise Names* | *Inventory Sets* | *Process Flows* | *Distribution Networks* | *Responsibility Assignments* | *Timing Cycles* | *Motivation Intentions* | |

*Horizontal integration lines are shown for example purposes only and are not a complete set. Composite, integrative relationships connecting every cell horizontally potentially exist.

# Framework Graphic

For the latest version of the Framework Graphic,
register at **www.Zachman.com**
for a high resolution .pdf file.

(For a publication release of the Framework Graphic
send requests to the Contact Us link on zachman.com)

You may be interested in several articles by John A. Zachman at
Zachman.com
"Architecture Is Architecture Is Architecture"
"John Zachman's Concise Definition of the Zachman Framework"
and
"The Zachman Framework Evolution" by John P. Zachman

# Engineering vs Manufacturing

Engineering work requires

**single-variable,**

(Synthesis)

ontologically-defined descriptions

of the ***whole*** of the object.

(Primitive)

(This is RADICALLY different)

**IN CONTRAST**

Manufacturing work requires

**multi-variable,**

holistic descriptions

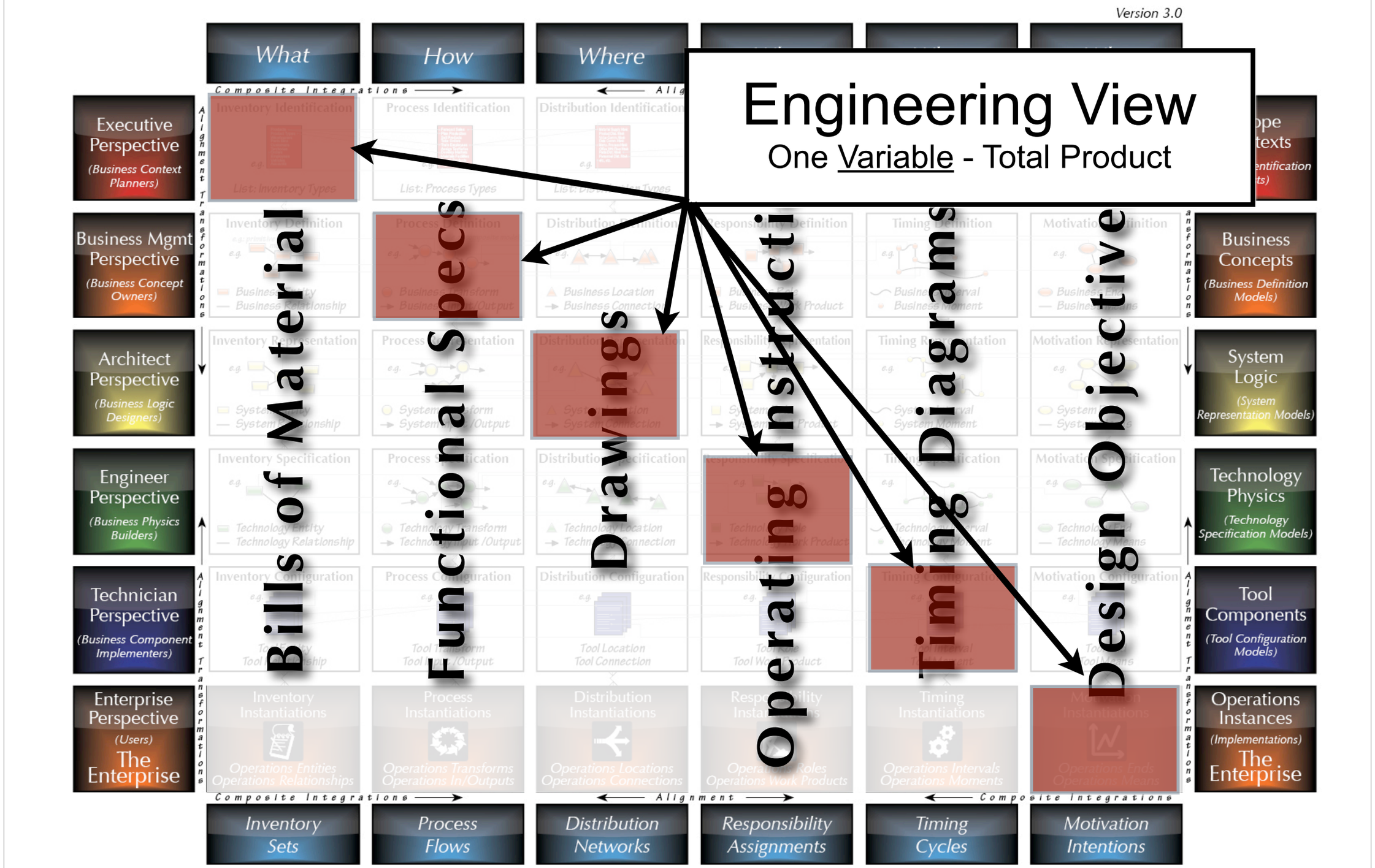
of ***parts*** of the object.

(Analysis)

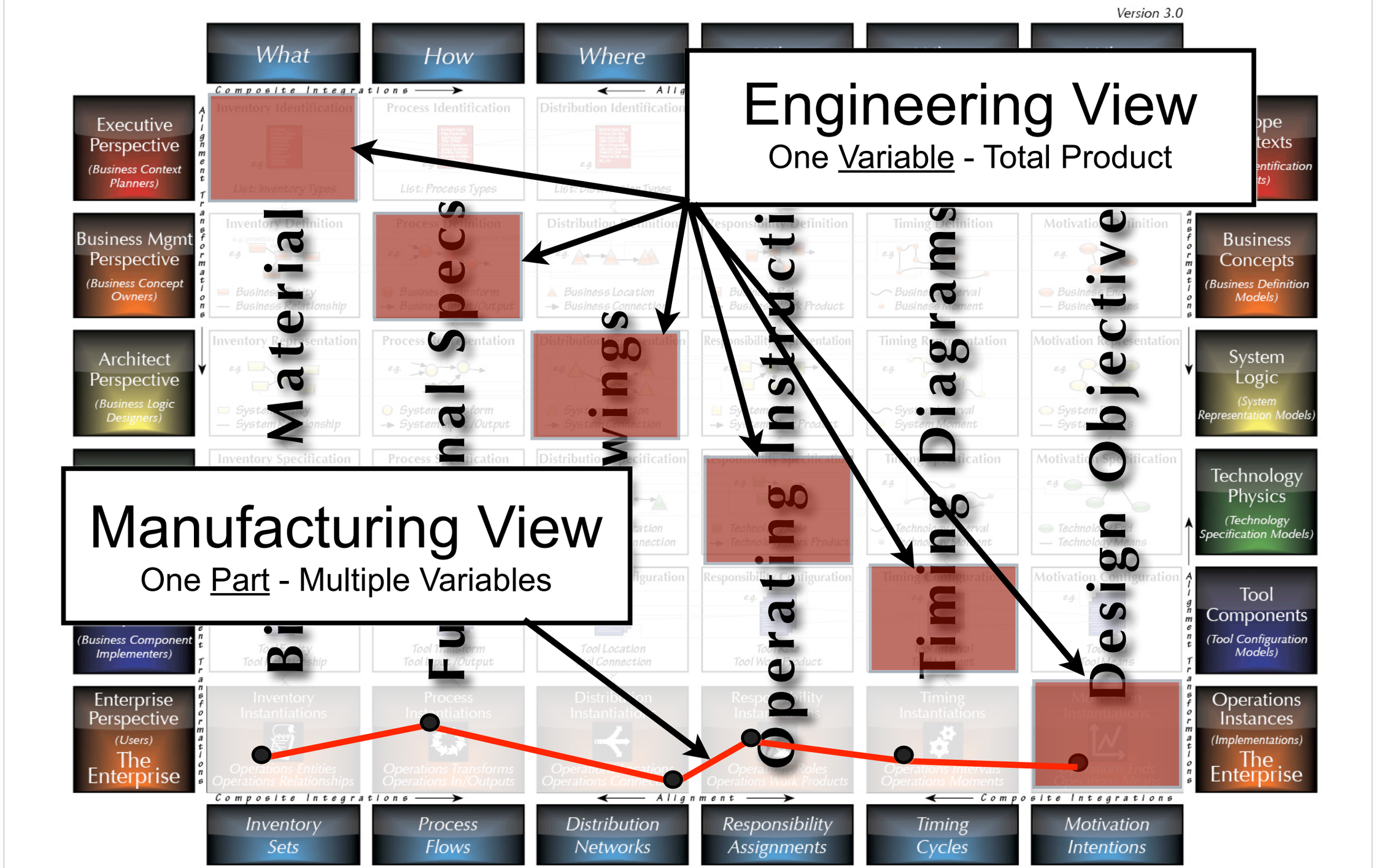
(Composite)

(This is STANDARD practice)

# Engineering Versus Manufacturing



# Engineering Versus Manufacturing



# 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).

A Framework is a STRUCTURE.
(A Structure DEFINES something.)

---

# Methodology

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

---

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

# Ontology vs Methodology

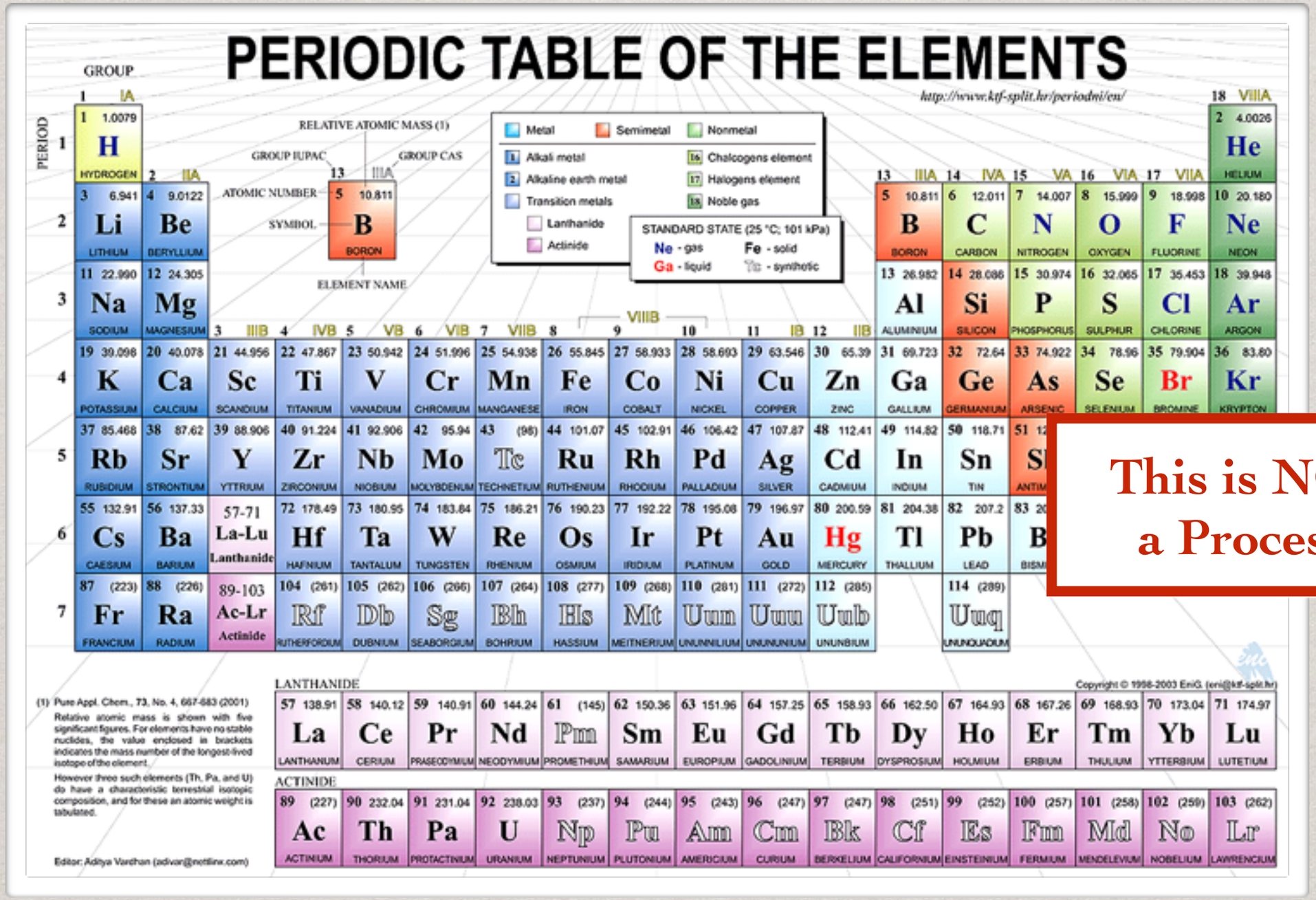
An Ontology is the classification of the total set of "**Primitive**" (elemental) components that exist and that are relevant to the existence of an object.

A Methodology produces "**Composite**" (compound) implementations of the Primitives.

**Primitives (elements) are timeless.**

**Composites (compounds) are temporal.**

# Ontology




This is NOT a Process.

Elements are Timeless

Until an ontology exists, nothing is repeatable, nothing is predictable.
There is no DISCIPLINE.

# Process

(Methodology)

A Process TRANSFORMS something.

This is a Process:

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

This is NOT an Ontology.

Compounds are Temporal

# Process

## (Methodology)

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

$$HCl + NaOH \rightarrow NaCl + H_2O$$

## Compounds

| | |
|---|---|
| Salt | NaCl |
| Aspirin | $C_9H_8O_4$ |
| Vicodin | $C_{18}H_{21}NO_3$ |
| Naproxen | $C_{14}H_{14}O_3$ |
| Ibuprophen | $C_{13}H_{18}O_2$ |
| Viagra | $C_{22}H_{30}N_6O_4S$ |
| Sulphuric Acid | $H_2SO_4$ |
| Water | $H_2O$ |

etc., etc., etc.

Compounds are Temporal

# Alchemy - A Practice

## This is a Methodology WITHOUT an Ontology

A Process with no ontological structure is ad hoc, fixed and dependent on practitioner skills.

This is NOT a science.

It is ALCHEMY, a "**practice**."

# Ontology

## The Zachman Framework for Enterprise Architecture™

### *The Enterprise Ontology* ™

Version 3.0

| Classification Names / Audience Perspectives | What | How | Where | Who | When | Why | Classification Names / Model Names |
|---|---|---|---|---|---|---|---|
| | Composite Integrations → | | ← Alignment → | | ← Composite Integrations | | |
| Executive Perspective (Business Context Planners) | Inventory Identification<br>e.g.<br>List: Inventory Types | Process Identification<br>e.g.<br>List: Process Types | Distribution Identification<br>e.g.<br>List: Distribution Types | Responsibility Identification<br>e.g.<br>List: Responsibility Types | Timing Identification<br>e.g.<br>List: Timing Types | Motivation Identification<br>e.g.<br>List: Motivation Types | Scope Contexts (Scope Identification Lists) |
| Business Mgmt Perspective (Business Concept Owners) | Inventory Definition<br>e.g.<br>Business Entity<br>Business Relationship | Process Definition<br>e.g.<br>Business Transform<br>Business Input/Output | Distribution Definition<br>e.g.<br>Business Location<br>Business Connection | Responsibility Definition<br>e.g.<br>Business Role<br>Business Work Product | Timing Definition<br>e.g.<br>Business Interval<br>Business Moment | Motivation Definition<br>e.g.<br>Business End<br>Business Means | Business Concepts (Business Definition Models) |
| Architect Perspective (Business Logic Designers) | Inventory Representation<br>e.g.<br>System Entity<br>System Relationship | Process Representation<br>e.g.<br>System Transform<br>System Input /Output | Distribution Representation<br>e.g.<br>System Location<br>System Connection | Responsibility Representation<br>e.g.<br>System Role<br>System Work Product | Timing Representation<br>e.g.<br>System Interval<br>System Moment | Motivation Representation<br>e.g.<br>System End<br>System Means | System Logic (System Representation Models) |
| Engineer Perspective (Business Physics Builders) | Inventory Specification<br>e.g.<br>Technology Entity<br>Technology Relationship | Process Specification<br>e.g.<br>Technology Transform<br>Technology Input /Output | Distribution Specification<br>e.g.<br>Technology Location<br>Technology Connection | Responsibility Specification<br>e.g.<br>Technology Role<br>Technology Work Product | Timing Specification<br>e.g.<br>Technology Interval<br>Technology Moment | Motivation Specification<br>e.g.<br>Technology End<br>Technology Means | Technology Physics (Technology Specification Models) |
| Technician Perspective (Business Component Implementers) | Inventory Configuration<br>e.g.<br>Tool Entity<br>Tool Relationship | Process Configuration<br>e.g.<br>Tool Transform<br>Tool Input /Output | Distribution Configuration<br>e.g.<br>Tool Location<br>Tool Connection | Responsibility Configuration<br>e.g.<br>Tool Role<br>Tool Work Product | Timing Configuration<br>e.g.<br>Tool Interval<br>Tool Moment | Motivation Configuration<br>e.g.<br>Tool End<br>Tool Means | Tool Components (Tool Configuration Models) |
| Enterprise Perspective (Users) The Enterprise | Inventory Instantiations<br>Operations Entities<br>Operations Relationships | Process Instantiations<br>Operations Transforms<br>Operations In/Outputs | Distribution Instantiations<br>Operations Locations<br>Operations Connections | Responsibility Instantiations<br>Operations Roles<br>Operations Work Products | Timing Instantiations<br>Operations Intervals<br>Operations Moments | Motivation Instantiations<br>Operations Ends<br>Operations Means | Operations Instances (Implementations) The Enterprise |
| Audience Perspectives / Enterprise Names | Inventory Sets | Process Flows | Distribution Networks | Responsibility Assignments | Timing Cycles | Motivation Intentions | |

*Horizontal integration lines are shown for example purposes only and are not a complete set. Composite, integrative relationships connecting every cell horizontally potentially exist.



"Primitives" are Timeless.

**Until an ontology exists, nothing is repeatable, nothing is predictable.**

**There is no DISCIPLINE.**

# Process

(methodology)

# Composites

(compounds)

COBOL Programs
Objects
BPMN Models
Swimlanes
Business Architecture
Capabilities
Mobility
Applications
Data Models
Security Architecture
Services

COTS
Technology Architecture
Big Data
Missions/Visions
Agile Code
Business Processes
DoDAF Models
Balanced Scorecard
Clouds
I.B. Watson
TOGAF Artifacts

**Etc., etc., etc.**

Compounds are Temporal

Introduction to
Enterprise Architecture

# A
# Zachman Framework
# Story

John A. Zachman
Zachman International

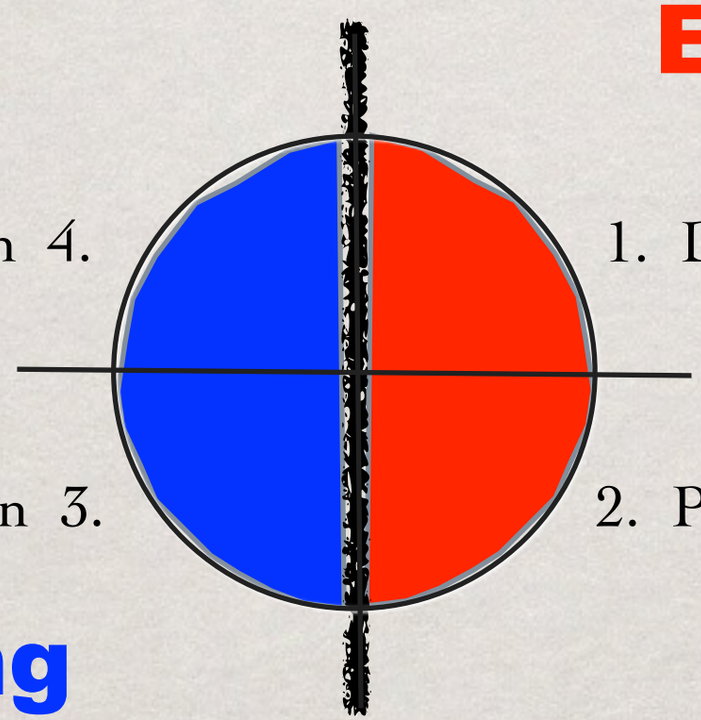
# Professional Service Cycle

**Engineering**

**(Ontology)**

Evaluation 4.

1. Diagnosis/Analysis of Need



Application/Implementation 3.

2. Prescription/Recommendation

**Manufacturing**

**(Methodology)**

Roger Greer:
Dean
School of Library and Information Management
University of Southern California
(My notes from a 1991, IBM GUIDE Conference presentation)

Introduction to
Enterprise Architecture

# Illustrations of Primitive Modeling Concepts

John A. Zachman
Zachman International

iCMG™

Enterprise Architecture Firm

| What | How | Where | Who | When | Why |
|---|---|---|---|---|---|
| Ex Pe (Bus Pla xt | Pro ication L ypes | Distr tification Lis n Types | Respons dentification Lis ity Types | Tim l | Moti on e.g. List: pes |
| Busin Agmt Pe (Bu pt | Pro ion e.g. Bus Bus | Distri nition e.g. Bu Bu ction | | | |
| A t P e (B sic D rs) | Invento tion e.g. Sys Sys ip | Proc e.g. S t | D on | Tin tion e.g. | M |
| F r P e (Bu sics Bu rs) | Inv ship | Pro ca STOP nsfo t /O put | Distribut cation e.g. Tech Tech tion | Respons ication e.g. Tech Tech Product | Timing S fication e.g. Tec Tec | Moti ion e.g. 1 1 |
| Technician P e (Bus nent In s) | Inv on | Proce To ut | Dist n Configuration | Re on | | Mo tion |
| Ent rise P e Enterprise | | Pro ar s | O Op s on | Tim | | |

Aricent_App_Rationalization - Base File - Enterprise Architect

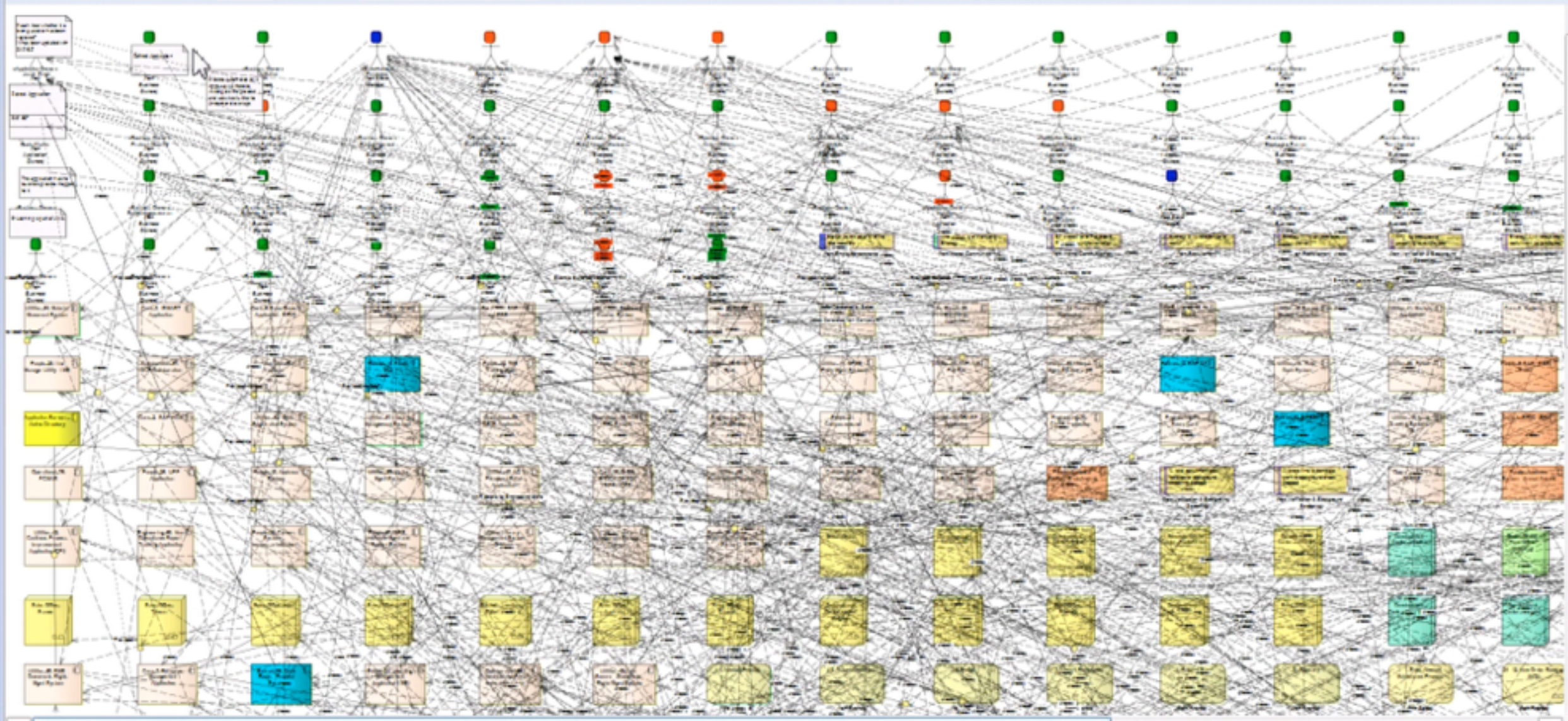
File Edit View Project Diagram Element Tools Analyzer Extensions Settings Window Help

BPMN™ 2.0 | Basic - Normal

Custom Diagram: "Moving Apps to DR Site - Impact Analysis - L5" created: 10/25/2011 6:11:15 PM modified: 8/1/2012 6:17:03 PM 38% 850 x 1098

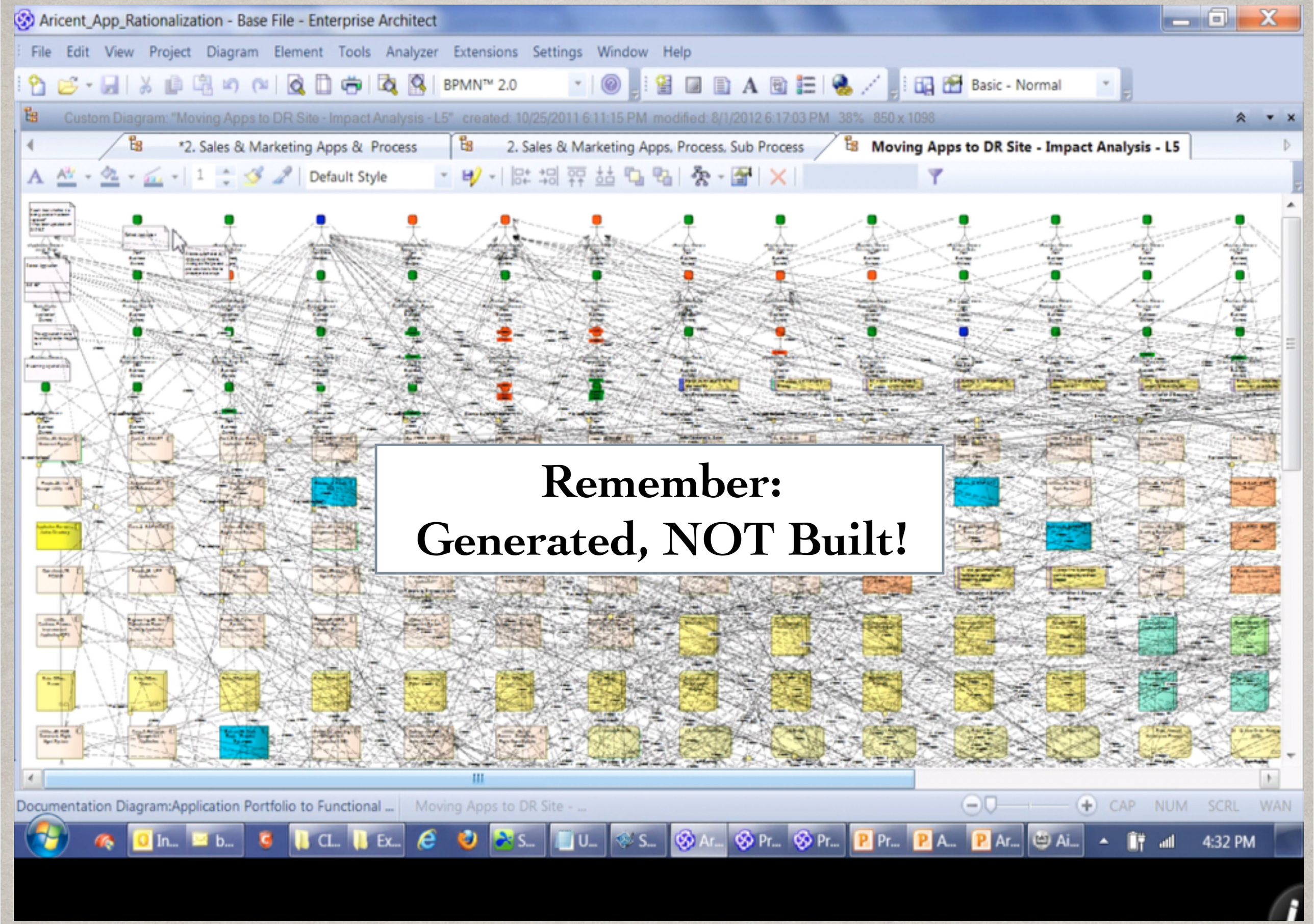
*2. Sales & Marketing Apps & Process | 2. Sales & Marketing Apps. Process. Sub Process | **Moving Apps to DR Site - Impact Analysis - L5**

Default Style



Documentation Diagram:Application Portfolio to Functional ... | Moving Apps to DR Site - ... | CAP NUM SCRL WAN

4:32 PM


Aricent_App_Rationalization - Base File - Enterprise Architect
File Edit View Project Diagram Element Tools Analyzer Extensions Settings Window Help
BPMN™ 2.0
Basic - Normal
Custom Diagram: "Moving Apps to DR Site - Impact Analysis - L5" created: 10/25/2011 6:11:15 PM modified: 8/1/2012 6:17:03 PM 38% 850 x 1098
*2. Sales & Marketing Apps & Process
2. Sales & Marketing Apps. Process. Sub Process
Moving Apps to DR Site - Impact Analysis - L5
Default Style
Remember:
Generated, NOT Built!
Documentation Diagram:Application Portfolio to Functional ...
Moving Apps to DR Site - ...
CAP NUM SCRL WAN
4:32 PM

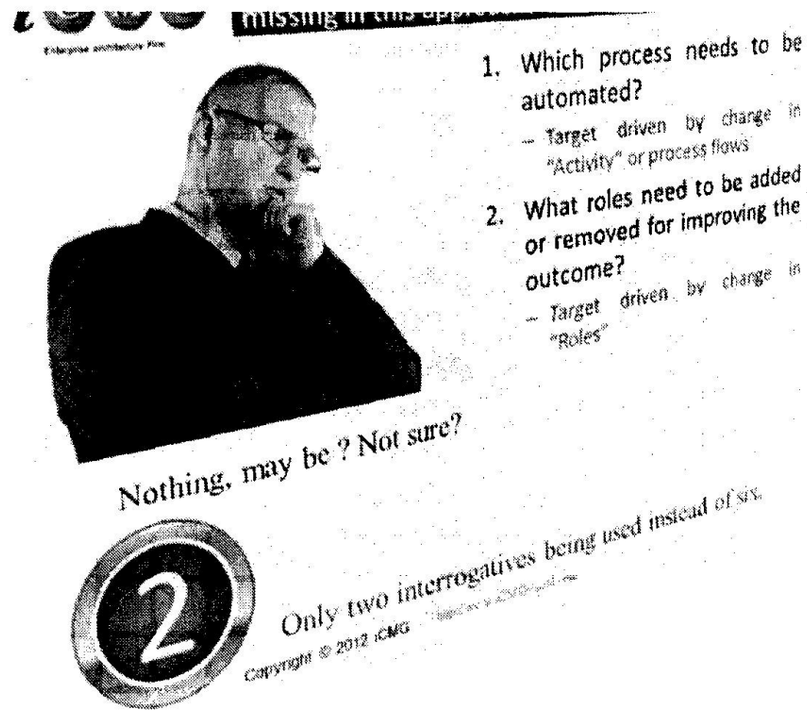
# The Key

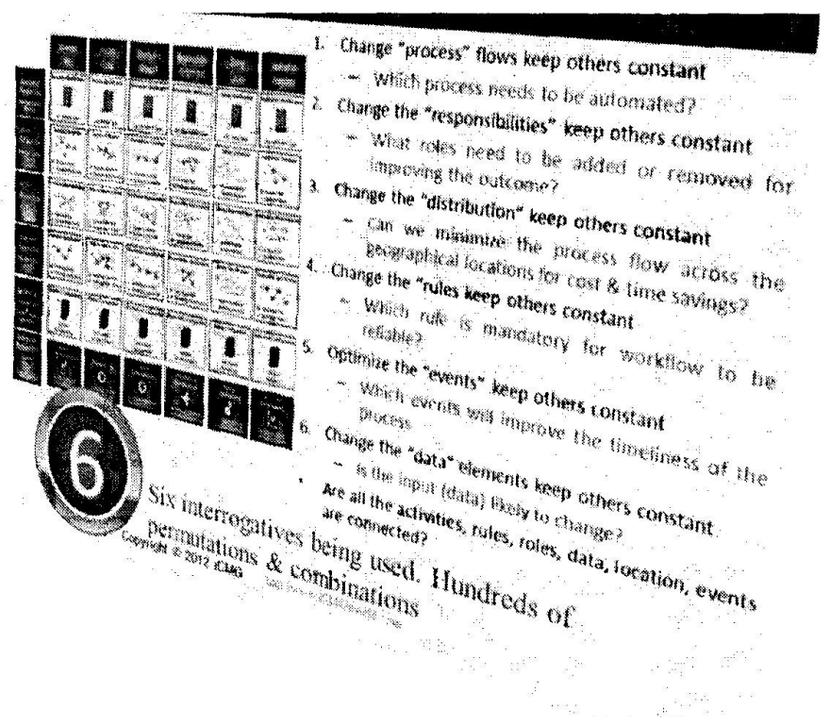
1. Single-variable, precisely unique, relevant (not arbitrary), ontologically-based components.

2. Binary Relationships (only two components at a time).

Enterprise Architecture Firm

# Exercise - Deconstruct Business Process Model for multiple Targets

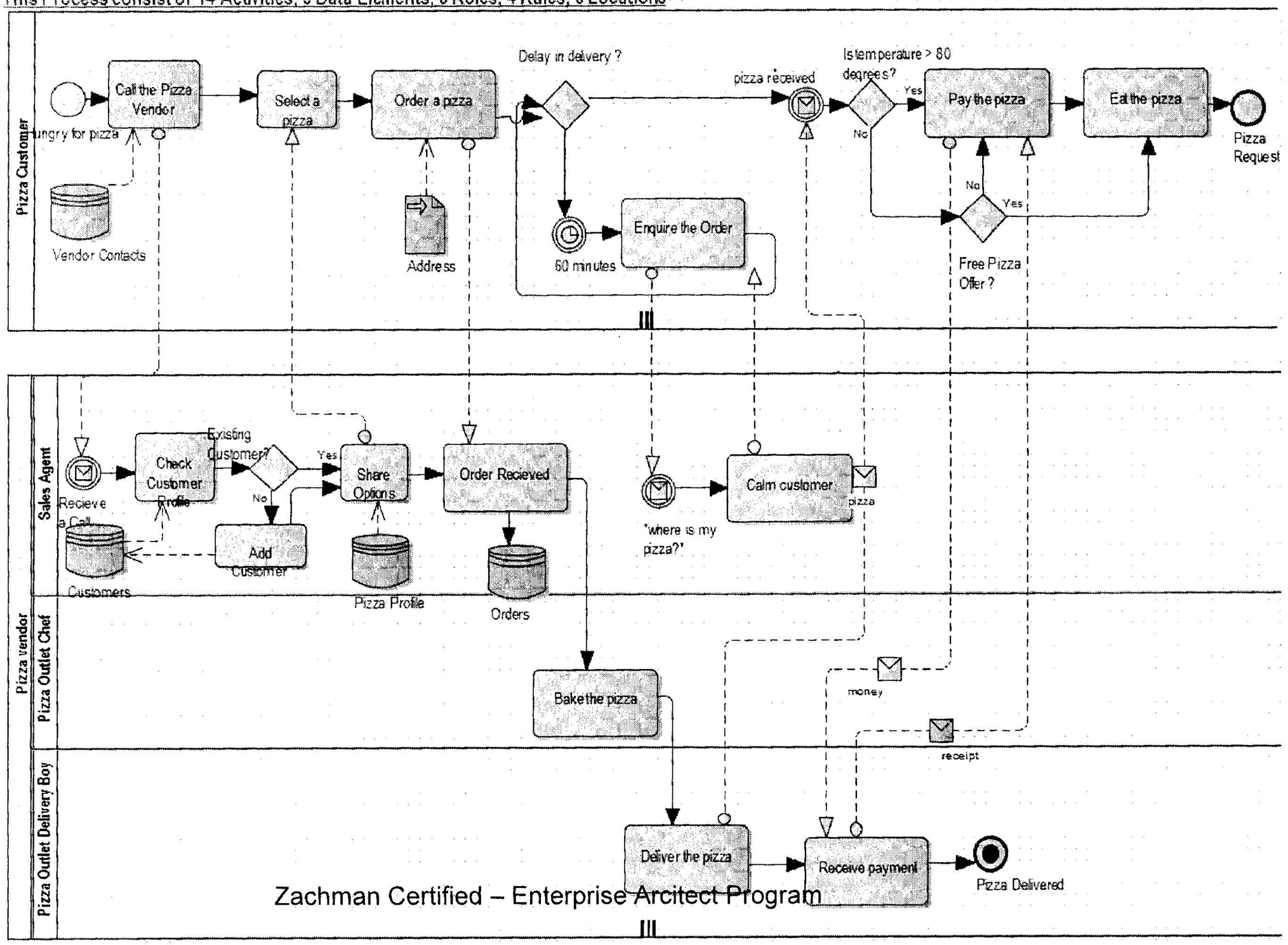


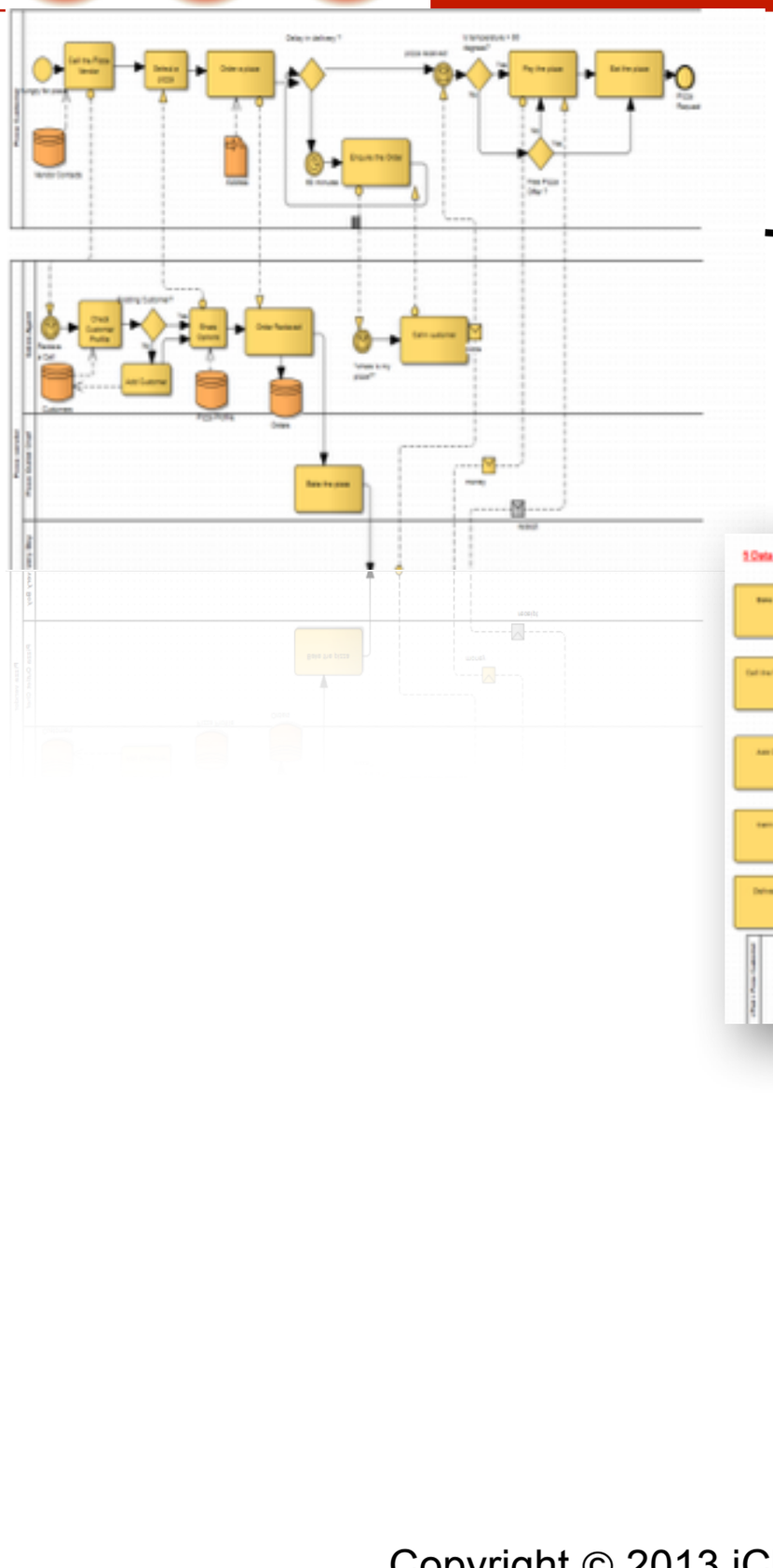

Zachman Certified – Enterprise Arcitect Program

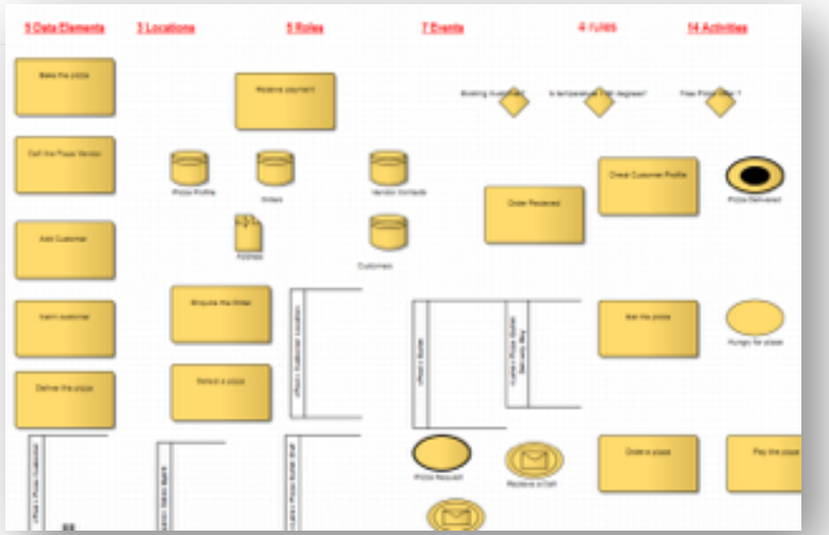
# Exercise – Pizza Delivery Process

This Process consist of 14 Activities, 5 Data Elements, 5 Roles, 4 Rules, 3 Locations

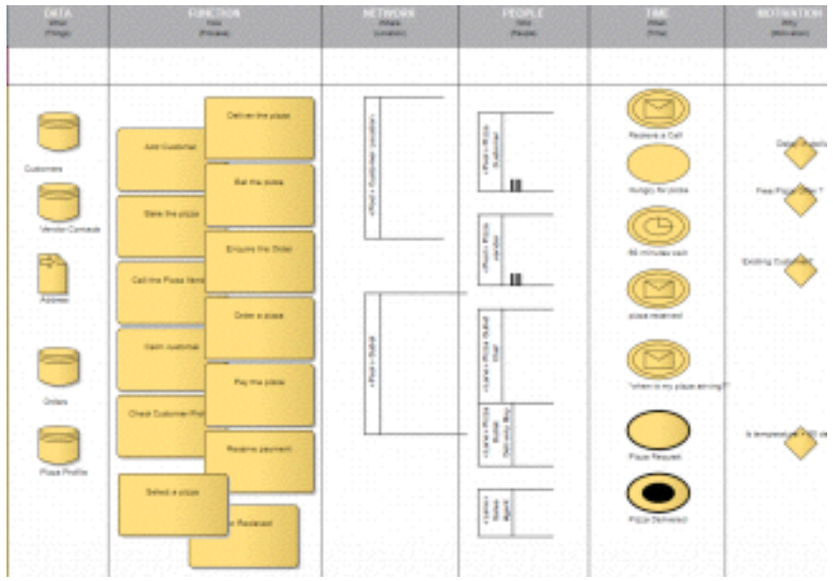


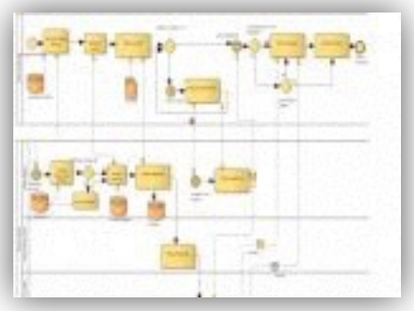
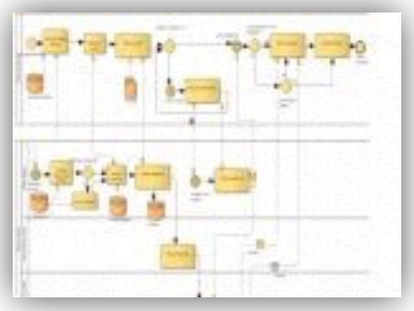
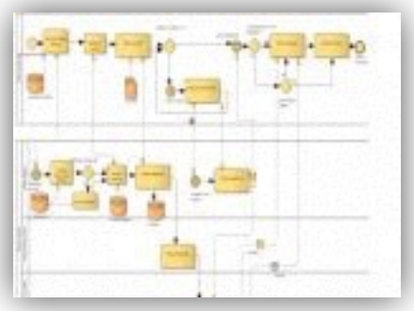
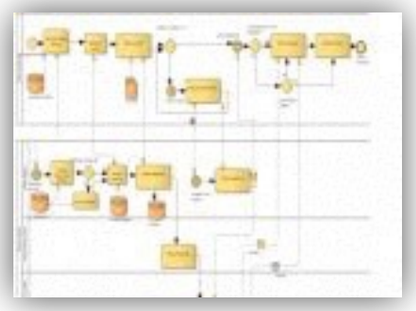
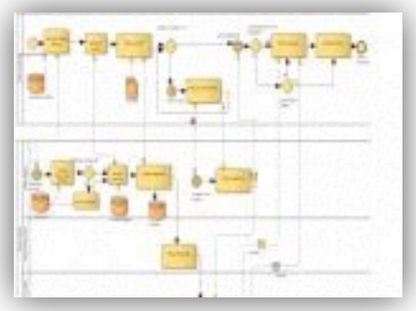
# Exercise 3

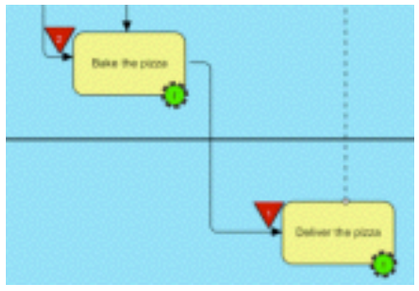


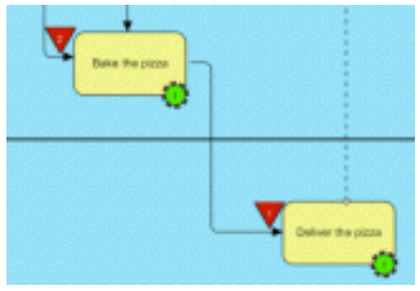


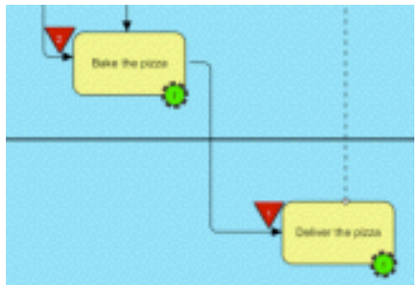

Exercise 3

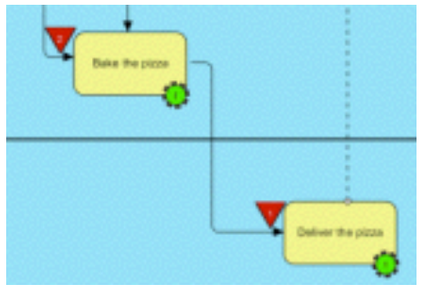


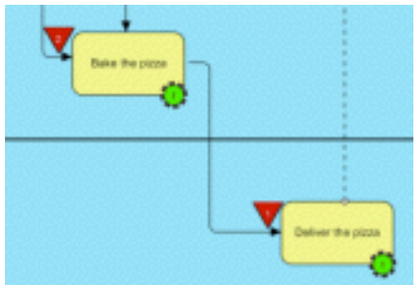







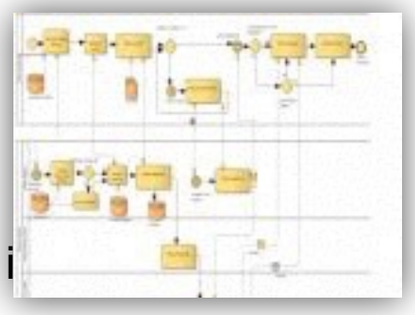






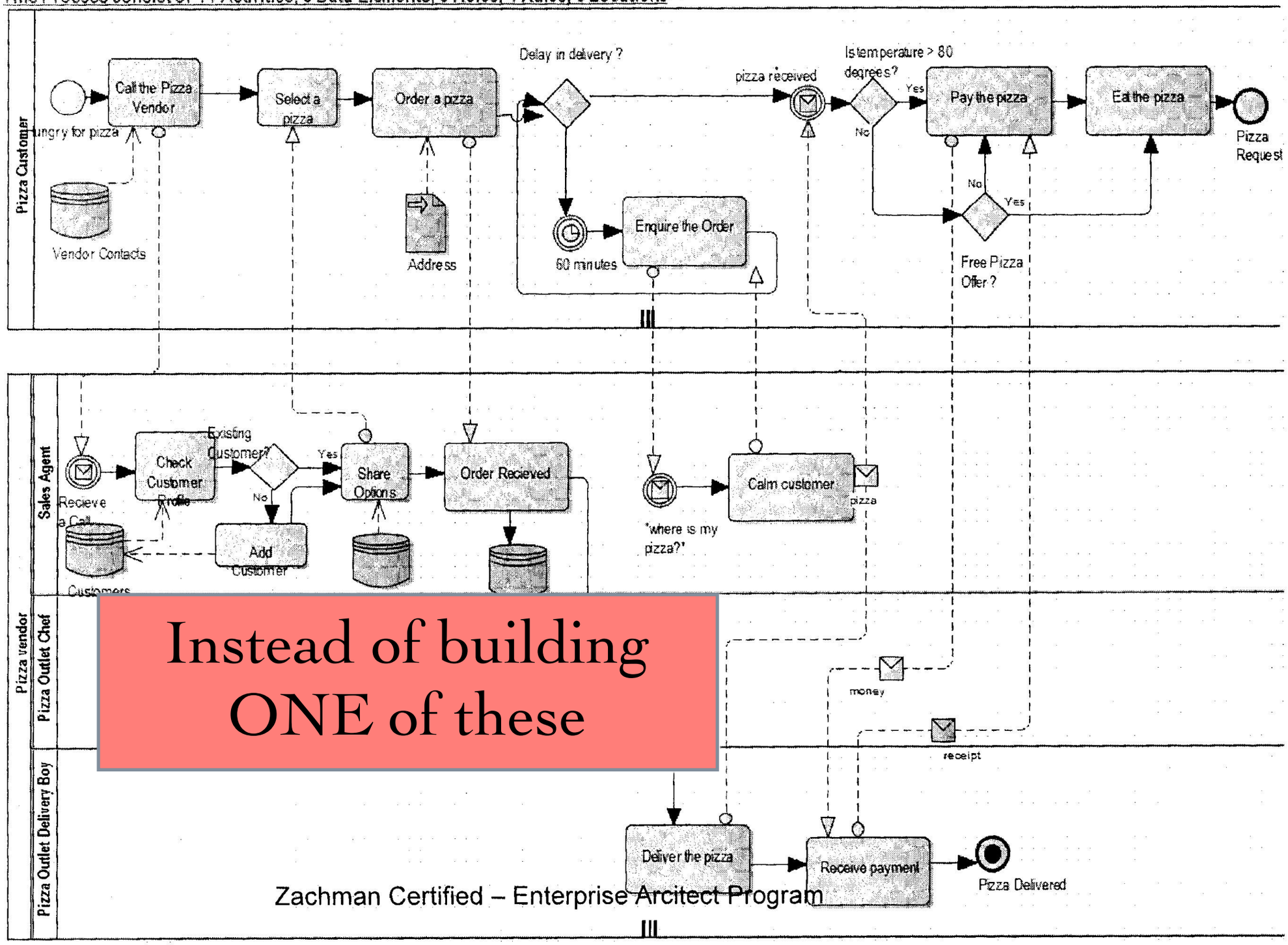




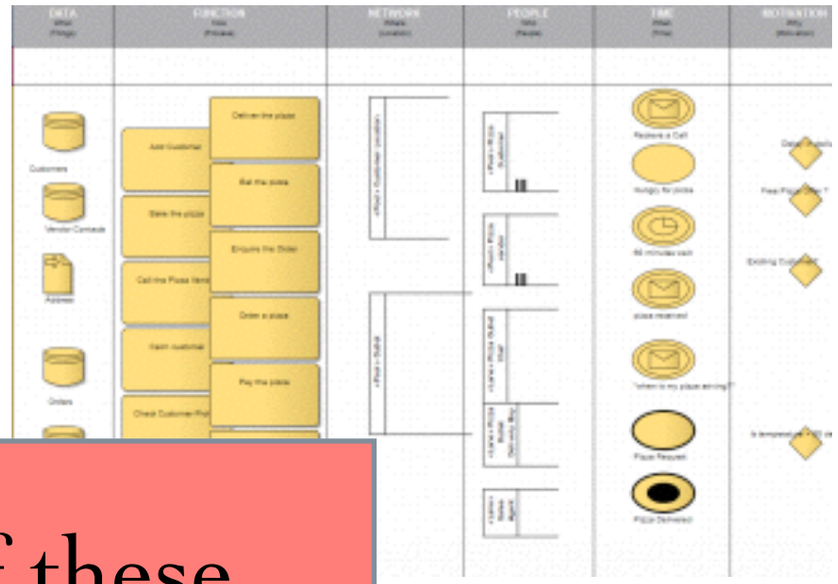
# Exercise – Pizza Delivery Process

This Process consist of 14 Activities, 5 Data Elements, 5 Roles, 4 Rules, 3 Locations

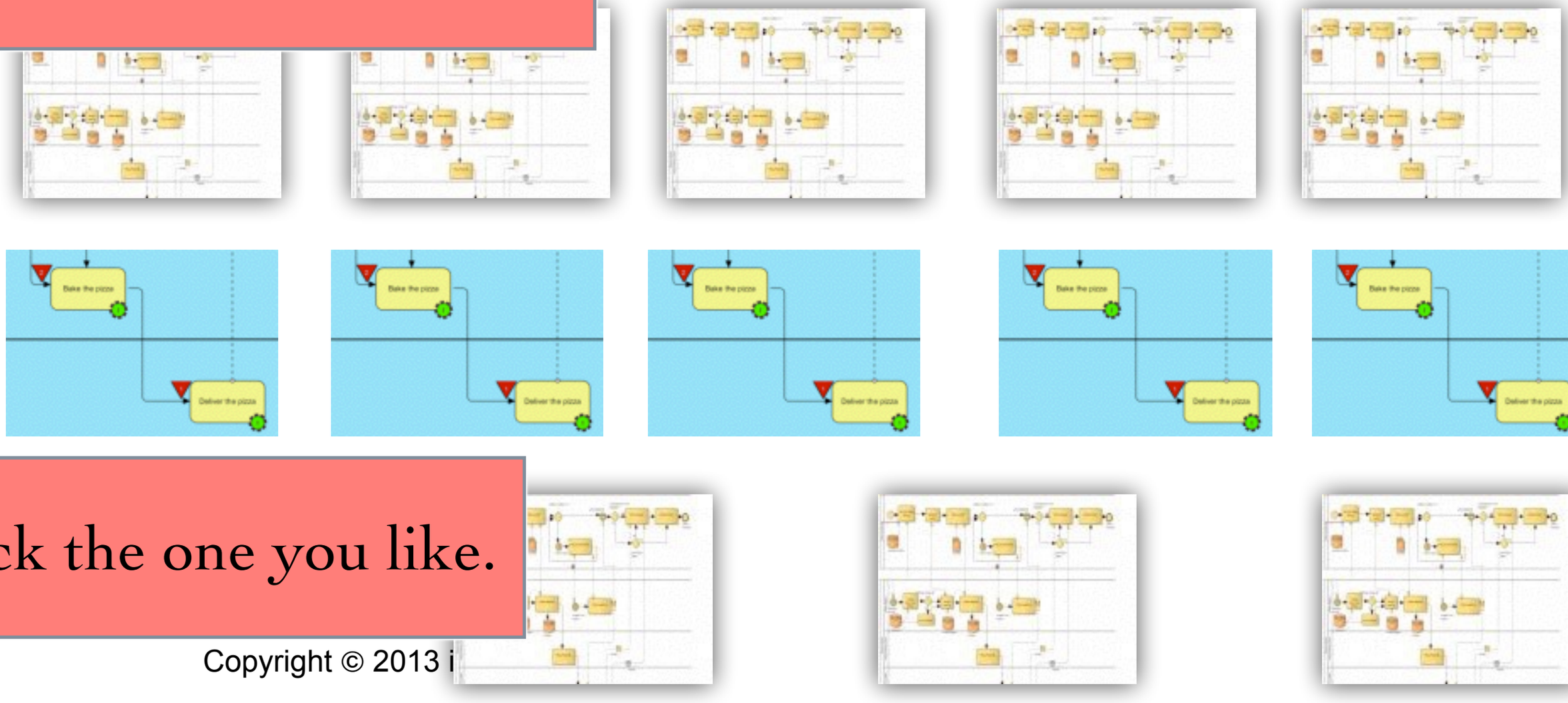



Instead of building ONE of these

Exercise 3



Build “n” of these.



Pick the one you like.

# The Key

1. Single-variable, precisely unique, relevant (not arbitrary), ontologically-based components.

2. Binary Relationships (only two components at a time).

Remember! This is a PRIMITIVE (single-variable) Model used for Engineering.

It cannot be used for implementations which require COMPOSITE (multi-variable) Models.

(Some possible COMPOSITE integration relationships may be shown at the periphery of the model. The COMPOSITE implementation "view" would be created by re-using components of other Enterprise-wide, "engineered" PRIMITIVES.

# What
## (Column 1)
## Inventory Identification

Row 1
Executive
Perspective

Note:
Air Transportation Case
Inventories (Entities)
Countable Things (Nouns)
(Likely have serial numbers)
A List - Scope
Level of Detail = High
Abstract (no instances)
As simple as Possible
No Recurring Concepts

| |
|---|
| Airplanes |
| Airplane Types |
| Airports |
| Gates |
| Passengers |
| Seats |
| Bookings |
| Employees |
| Vehicles |
| Routes |
| Flights |
| etc. |

Row 6 Instances AS IS may or may not have anything to do with Owner's, Designer's, Builders perceptions until those are made explicit and transformed to Row 6.

Inventory
Sets

Scope
Contexts

Composites
There can be composite relationships with any or all other Row 1 Cells and with the Cell below and Instances in Row 6.

Note: This sample model is meant to illustrate the form of the expected Primitive, not necessarily the content.

# Enterprise Framework

Remember! This is a PRIMITIVE (single-variable) Model used for Engineering.
It cannot be used for implementations which require COMPOSITE (multi-variable) Models.
(Some possible COMPOSITE integration relationships may be shown at the periphery of the model. The COMPOSITE implementation "view" would be created by re-using components of other Enterprise-wide, "engineered" PRIMITIVES.

How
(Column 2)
Process Identification

Row 1
Executive
Perspective

| |
|---|
| Acquire Routes |
| Schedule Flights |
| Sell Bookings |
| Reserve Seats |
| Train Employees |
| Fly Airplanes |
| Schedule Crews |
| Repair Facilities |
| Develop Markets |
| Maintain Airplanes |
| etc. |

Scope
Contexts

Note:
Air Transportation Case
Processes (Transformations)
(Transitive Verb-Object)
A List - Scope
Level of Detail = High
Abstract (no instances)
As simple as possible
No Recurring Concepts

Composites
There can be composite relationships with any or all other Row 1 Cells and with the Cell below and Instances in Row 6.

Row 6 Instances AS IS may or may not have anything to do with Owner's, Designer's, Builders perceptions until those are made explicit and transformed to Row 6.

Note: This sample model is meant to illustrate the form of the expected Primitive, not necessarily the content.

Process
Flows

| Inventory Sets | Process Flows | Distr. Networks |
|---|---|---|
| Airplanes | Acquire Routes | Airplane Network |
| Airplane Types | Schedule Flights | Parts Distr. Net. |
| Airports | Reserve Seats | Communications |
| Gates | Train Employees | Freight Net. |
| Passengers | Fly Airplanes | Airport Network |
| Shareholders | Schedule Crews | (Runways, etc.) |
| Local Carriers | Repair Facilities | Regulatory Net. |
| Seats | Develop Markets | Passenger Net. |
| Bookings | Maint. Airplanes | Personnel Net. |
| Routes | Load Airplanes | Catering Net. |
| Employees | Release Flights | etc. |
| Vehicles | Develop Flt. Plans | |
| Flights | ScheduleMaint. | |
| etc. | etc. | |

| Respon Assmts | Timing Cycles | Motive Intent. |
|---|---|---|
| Pilots | Flight Cycle | Equip. Utilization |
| Co-pilots | Customer Cycle | New Markets |
| Engineers | Maintenance Cyc. | Revenue Growth |
| Flt. Attend. | Telephone Wait C. | Exp. Reduction |
| Reservations | Airplane Turnaround | Cust Convenience |
| Aircraft Maint. | De-Icing Cycle | Cust. Satisfaction |
| Flight Scheduling | Air Traffic Cntl. C. | Labor Contracts |
| Airport Ops.Mgt | Tarmac Cycle | Regulatory Comp |
| Customer Service | Airplane Cycle | New Capital |
| Marketing | Baggage Handling C. | Load Factor |
| Sales | Security (TSA)Cycle | Route Optimize |
| Flight Dispatch | Planning Cycle | Flight Expansion |
| Accounting | Budget Cycle | Acquisition |
| etc. | etc. | etc. |

# The Key

1. Single-variable, precisely unique, relevant (not arbitrary), ontologically-based components.

2. Binary Relationships (only two components at a time).

Introduction to
Enterprise Architecture

# Methodology for Solving General Management Problems

John A. Zachman
Zachman International

# The Process

1. Select General Management Problem.

2. Factor out Primitive (**Single-variable**) Components, sort into Zachman Framework Cells (make Lists).

3. Define "binary" (only two at a time) dependencies (horizontal and vertical) between Primitive (**Single-variable**) Components.

4. Create Composite (**Multi-varible**) "snapshot" of problem area and diagnose.

5. Pose new Composite (**Multi-variable**) scenarios (change Lists and/or change dependencies) and re-compose multiple targets.

6. Add time and cost to Primitive (**Single-variable**) Components and simulate alternatives, perform risk analysis, identify resource rqmts, etc.

(CEO picks solution, assigns responsibilities for "quick-fix", identifies subsequent CEO problem. Then re-iterate Steps 1 - 6.)

# The Process (cont.)

(Architecture process proceeds in parallel:)

7. Pick several Cells in different Columns in some Row and assign modeling experts to build out complete (thing-relationship-thing) Primitive (**Single-variable**) Models, verifying horizontal alignment.

8. Create complete Composite (**Multi-variable**) integration for Row ensuring horizontal alignment. (Does each Cell have all components required for reuse in adjoining Cells for creating Composites.)

9. Have Columnar modeling experts transform Primitive (**Single-variable**) Models to next Cell below, ensuring vertical "alignment" and iterate until all Cells in the Column are transformed and vertically aligned.

10. Transform Row 5 Primitives (**Single-variables**) to Row 6 implementations (**Multi-variables**)using either machines (automated) or people (manual).

11. Add Primitive Components by Cell from next problem and reiterate Steps *7* - 11.

# The Process (cont.)

12. Institutionalize this process and govern Architecture as follows:
    a. prohibit redundancy except where explicitly controlled.
    b. maintain horizontal and vertical alignment
    c. use Primitive (**Single-variable**) Model inventory as base for managing **ENTERPRISE** changes.
    d. ensure **EVERY** new implementation Composite reuses components of Primitive models and migrate legacy to Architected Enterprise. (See Workshop "Migration Strategy".)
13. Acquire subject matter expertise for building additional Primitive (**Single-variable**) Models to be added to the Enterprise Architecture capability inventory.

**Key: Single-Variable, PRIMITIVE Models, and Binary Relationships**

For details see Level 2 Zachman Certification at www.Zachman.com

Note: This is the same process, somewhat abbreviated, and executed by students in the 4 day Zachman Level 1 Certification Workshop.

Enterprise Architecture

# Conclusions

John A. Zachman
Zachman International

# Research Lessons

A. It is possible to solve General Management problems very quickly with a small subset of Primitive components

(simply Lists and their inter-dependencies

short of the complete Primitive Models)

B. Different complex, composite constructs can be created dynamically, virtually cost-free, from the inventory of Primitive Lists for addressing subsequent General Management problems.

C. Many scenarios can be evaluated to test strategy alternatives before making commitments.

# Profound Significance

A. It alters the concept of Enterprise Architecture from one of building models to one of solving General Management problems.

B. Proves the validity of the Primitive Model concept: from a finite inventory of Primitive Concepts you can dynamically create a virtually infinite number of Enterprise implementation Composites.

C. Buys the time for "the experts" to build out the complete Enterprise Architecture (Thing-Relationship-Thing) Primitive Models iteratively and incrementally.

D. Builds significant credibility for the Information Technology community.

E. Establishes the basis for an Enterprise Architecture Profession.

# Challenge to Enterprise Architects

Reframe the concept of Enterprise Architecture ...

It is not about building models!

**It is about solving Enterprise problems while iteratively and incrementally building out the inventory of complete, reusable, Primitive Models that constitute:**

**Enterprise Architecture.**

# Jay W. Forrester

"Although social systems are more complex than physical systems, they belong to the same class of high-order, non-linear, feedback systems as do physical systems.

People do not accept the idea that families, corporations, and governments belong to the same class of dynamic structures as do chemical refineries and autopilots for aircraft.

"Organizations built by committee and intuition perform no better than would an airplane built by the same methods ... As in a bad airplane design, which no pilot can fly successfully, such badly designed corporations lie beyond the ability of real-life managers.

"I anticipate future management schools devoted to 'enterprise design'. ... A fundamental difference exists between an enterprise operator and an enterprise designer. A manager runs an organization, just as a pilot runs an airplane. Success of a pilot depends on an aircraft designer who created a successful airplane. ...who designed the corporation that a manager runs?"

*"Designing the Future" by Jay W. Forrester 12/15/98*

# 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)*

# 2015 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. 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, Casewise, 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, Agile Programming, Service Oriented Architecture, Cloud Computing, BigData or I.B.Watson (whoever that is) is going to fix the problems.

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

# 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 problems. My plan would be to start building out an inventory of models, PRIMITIVE MODELS, iteratively and incrementally, engineering them for alignment, integration, flexibility, reduced time-to-market, etc., etc.

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