Analyzing your project in order to build a database and a use system

Anthony Aristar
Most linguists build their projects by starting from the lowest end, e.g.:

- I need to get working! How do I get the answers I need quickly?
  
  Or

- I need to store this data, where can I stick it while I work on the really important stuff?

This produces systems which:

- Are in constant flux
- Can never produce the data in the way that it is really needed.
- Worst of all, look like they were just thrown together. (Hmmm, they were, weren’t they?)
There’s a better way, but it takes some self-discipline to use it.

That way is to grit your teeth and take the time to analyze all aspects of your project and plan your information system (database and other systems) before you do anything.

There are techniques that allow you to do that... and they pay off! In computerese, it’s called “Information Engineering.”
... is a formal methodology that is used to create and maintain information systems.

... and always starts with the **project model** and works in a **Top-Down** fashion to build supporting data models and process models which best suit your project.

**Basic rule:** *Do the hard parts first!*
Planning

Analysis

Design

Implementation
- 1987, Zachman, IBM Systems Journal
- 1993, Sowa and Zachman, IBM Systems Journal
- A popular project view of the components of an information system and the process of developing it.
- But equally useful in academic systems because it helps you design an information system of *any* type.
- This is the design that underlies all the systems used by the LINGUIST List.
In the beginning…

- This all started because someone noticed that you could build a house or a battleship on time and in budget, but software projects never...

- So IBM brought in a guy named Zachman who analyzed what was going on.

- He noted that the later mistakes were caught, the longer they took to fix, and the more money they cost.
### The Full Zachman Framework

<table>
<thead>
<tr>
<th>DATA</th>
<th>FUNCTION</th>
<th>NETWORK</th>
<th>PEOPLE</th>
<th>TIME</th>
<th>MOTIVATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>What</td>
<td>How</td>
<td>Where</td>
<td>Who</td>
<td>When</td>
<td>Why</td>
</tr>
<tr>
<td>List of Things Important to the Business</td>
<td>List of Processes of the Business Process</td>
<td>List of Locations in which the Business Operates</td>
<td>List of Organizations Important to the Business</td>
<td>List of Events Significant to the Business</td>
<td>List of Business Goals and Strategies</td>
</tr>
<tr>
<td>e.g., Semantic Model</td>
<td>e.g., Business Process Model</td>
<td>e.g., Logistics Network</td>
<td>e.g., Work Flow Model</td>
<td>e.g., Master Schedule</td>
<td>e.g., Business Plan</td>
</tr>
</tbody>
</table>

#### Abstractions Perspectives

**Scope**
- Planner
  - Contextual
  - Conceptual

**Enterprise Model**
- Owner
  - Logical
    - e.g., Logical Data Model
      - Entity = Business Entity
      - Process = Business Process
      - Node = Business Location
      - People = Organization Unit
      - Time = Business Event
      - End = Business Objective
      - Means = Business Strategy

**System Model**
- Designer
  - Physical
    - e.g., Physical Data Model
      - Entity = Data Entity
      - Process = Application Function
      - Node = IS Function
      - People = Role
      - Time = System Event
      - End = Structural Assertion
      - Means = Action Assertion

**Technology Constrained Model**
- Builder
  - e.g., System Design
    - e.g., Technical Architecture
      - e.g., Presentation Architecture
      - e.g., Control Structure
      - e.g., Rule Design

**Detailed Representations**
- Subcontractor out of context
  - e.g., Data Definition
    - e.g., Program
      - e.g., Network Architecture
      - e.g., Security Architecture
      - e.g., Timing Definition
      - e.g., Rule Specification

John A. Zachman, Zachman International
# A Simplified Model of the Zachman Framework

<table>
<thead>
<tr>
<th></th>
<th>Data (entities)</th>
<th>Process</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>project Scope</strong>&lt;br&gt;List of entities important to project</td>
<td>List of functions project performs</td>
<td>List of locations in which project operates</td>
</tr>
<tr>
<td>2</td>
<td><strong>project model</strong>&lt;br&gt;project entities &amp; their relationships</td>
<td>Function &amp; process decomposition</td>
<td>Comms. links between project locations</td>
</tr>
<tr>
<td>3</td>
<td><strong>ISs model</strong>&lt;br&gt;Model of project data &amp; their relationships</td>
<td>Flows between apps. processes</td>
<td>Distribution network</td>
</tr>
<tr>
<td>4</td>
<td><strong>Technology model</strong>&lt;br&gt;Database design</td>
<td>Process specification</td>
<td>Configuration design</td>
</tr>
<tr>
<td>5</td>
<td><strong>Technology definition</strong>&lt;br&gt;database schema &amp; subschema definition</td>
<td>Process Specification (e.g. export)</td>
<td>Configuration definition</td>
</tr>
<tr>
<td>6</td>
<td><strong>Information system</strong>&lt;br&gt;data &amp; information</td>
<td>Apps. programs</td>
<td>System configuration</td>
</tr>
</tbody>
</table>
Zachman modeled his enterprise architecture on the processes involved in building a house.

You don’t just hire a carpenter and give him some nails

You have at least 6 roles involved in the design of a house
Six roles involved in the design of a house

- **Owner** (defines Project scope)
- **Architect** (makes Project model)
- **Designer** (makes System model--chooses fittings, materials)
- **Builder** (makes Technology model--plans the actual building process)
- **Sub-contractor** (implements Technology--installs the plumbing, builds walls, etc.)
- **Resident** (makes use of the project)
Six Levels in Project Design

- How does this model translate into analyzing a linguistic project?
- Six levels at which the project can be analyzed:
  - Project scope: a strategic overview including project scope/mission & direction
  - Project model: what entities take part in the project and their relationships
  - Information Systems model: how this translates into a database design and what forms are necessary in order to access it in the ways needed.
  - Technology model: Converts IS model into a design that conforms to the features/constraints of the technology
  - Technology definition (Programmer): Converts technology models into statements to generate the actual IS
  - Information system (User): Manages, uses and operates the completed IS
The project can be broken down into 3 parts: Data (entities), Processes, Network (locations)

Each of these can be analyzed at all 6 levels
June 24-27, 2008

Data Management & Archiving

1. Project Scope (Overview, goals)
### Data (Entities)
- **People**
  - Researcher
  - Consultants
  - Technology consultant

- **Data**
  - Lexical items
  - Glosses

- **Media**
  - sound files
  - video files

- **Equipment**
  - computers…

### Process
- Collecting Data
- Analyzing data
- Storing data
- Publishing data

### Network
- Field setting (village)
- Field setting (town)
- Home office
- University office
**2. Project Model (Architect)**

*Data*  
- Project entities and their relationships

*Process*  
- Function and process decomposition

*Network*  
- Communications links between project locations

**Project model:** how do the entities interact, and where?
Data (Entities)
Recorders record one or more sound files...
Sound files contain one or more texts
Consultants are linked to sound files
Lexical items appear in one or many texts
Glosses appear on lexical items
video files contain texts

Process
Searches on lexical items need to be made by sub-string search, by whole words
You need to know what files a lexical item appears in, how often they appear, what their context is.
You need to know what texts a consultant provided, and at what location.

Network
Where are copies of the data kept?
What machines does each kind of file appear on? Where is that machine?
3. Information System Model (Designer)

How do we translate this into a database design?
InField

Information system design: Documentation project

Data (Entities)
Build database design based on project model: e.g. where lexical item can appear in more than one file, reflect it.

Where consultant provides many texts, reflect it.

Process
Define the set of forms which will be required to access the data according to the previous two steps, and how they will relate.

Network
Define where this data is to be distributed from.
4. Technology Constrained Model
(Builder)
<table>
<thead>
<tr>
<th>Data (Entities)</th>
<th>Process</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modify design if necessary to fit technology, assign tasks</td>
<td>Modify design if necessary to fit technology, assign tasks</td>
<td>If necessary, redesign database to fit.</td>
</tr>
</tbody>
</table>
Now let’s write the system...

5. Technology Definition/ Detailed Representations
   (Programmer)
<table>
<thead>
<tr>
<th>Data (Entities)</th>
<th>Process</th>
<th>Network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instantiate database</td>
<td>Write forms to access data</td>
<td>Instantiate network if needed.</td>
</tr>
</tbody>
</table>
6. Functioning project (User)

Data
- Implemented Database and information

Process
- Implemented Application Programs

Network
- Current System Configuration
- A fully functional database and access system.
Principles of Good Design

- Understand your project
- Use models to design your system
- Use hierarchical, top-down design, because when these change they force changes everywhere.
- Do the hard parts first
- Focus on high-risk entities first
- State what, not how (called polymorphism in object-oriented programming). State what needs to be done, not how to implement it (e.g. “play sound data”, not “run cd” or “run tape”).
- Allocate each function to only one component: too many cooks spoil the broth. Also, functions should not depend upon one another.
- Group data and behavior
- Prioritize
- Use open standards
- Your interface depends on your design, so design your interface last