Building a Functional Model

“How will the system work?”
Course Design

- Deciding What to Build & Why
- Bringing Solutions to Life
- Ensuring Systems Work And Are Robust
- Managing Evolution… Deciding What’s Next
Course Design

Building a Functional Model

Implementing the Functions

Specifying Components

Design Review

Bringing Solutions to Life
<table>
<thead>
<tr>
<th>Date</th>
<th>Week</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feb 1</td>
<td>1</td>
<td>• Thinking in Terms of Systems</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>Deciding What to Build and Why</em></td>
</tr>
<tr>
<td>Feb 8</td>
<td>2</td>
<td>• Defining the Problem</td>
</tr>
<tr>
<td>Feb 16</td>
<td>3</td>
<td>• Developing a Solution</td>
</tr>
<tr>
<td>Feb 22</td>
<td>4</td>
<td>• Formulating a Proposal</td>
</tr>
<tr>
<td>Feb 29</td>
<td>5</td>
<td>• Concept Review</td>
</tr>
<tr>
<td></td>
<td></td>
<td><em>Bringing Solutions to Life</em></td>
</tr>
<tr>
<td>Mar 7</td>
<td>6</td>
<td>• Building a Functional Model</td>
</tr>
<tr>
<td>Mar 14</td>
<td>7</td>
<td>• Implementing the Functions</td>
</tr>
<tr>
<td>Mar 28</td>
<td>8</td>
<td>• Specifying Components</td>
</tr>
<tr>
<td>Apr 4</td>
<td>9</td>
<td>• Design Review</td>
</tr>
</tbody>
</table>

© Copyright 2016 Stevens Institute of Technology, All rights reserved
<table>
<thead>
<tr>
<th>Date</th>
<th>Week</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 11</td>
<td>10</td>
<td><strong>Ensuring the System Works and Is Robust</strong></td>
</tr>
<tr>
<td>Apr 18</td>
<td>11</td>
<td>• Integration and Test</td>
</tr>
<tr>
<td>Apr 25</td>
<td>12</td>
<td>• Modeling and Simulation</td>
</tr>
<tr>
<td>May 2</td>
<td>13</td>
<td>• Designing for the Lifecycle</td>
</tr>
<tr>
<td>May 9</td>
<td>14</td>
<td>• Test Readiness Review</td>
</tr>
<tr>
<td>May 16</td>
<td>15</td>
<td><strong>Managing Evolution...Deciding What’s Next</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Technology and Innovation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• No Class - Final Project Submission</td>
</tr>
</tbody>
</table>
Deciding How the System Will Work

Key Questions:

1. What top-level functions will your system have to perform in order to accomplish it’s mission?

2. How else could you have partitioned the system functionality and why did you choose to partition it the way you did?

3. How will the top-level functions interact with each other and with their external environment?

4. How well will your functional architecture perform and what evidence can you provide to support that assessment?
Part 1: Introduction to System Architecture
Architecture as a Metaphor

• “…the systems engineer resembles an architect, who must generally have adequate substantive knowledge of building materials, construction methods, and so on, to ply his [or her] trade.

• “Like architecture, systems engineering is in some ways an art as well as a branch of engineering. Thus, aesthetic criteria are appropriate for it also. For example, such essentially aesthetic ideas as balance, proportion, proper relation of means to ends, and economy of means are all relevant in a systems-engineering discussion. Many of these ideas develop best through experience. They are among the reasons why an exact definition of systems engineering is so elusive.”

“The ideal architect should be a man [or woman] of letters, a skillful draftsman, a mathematician, a diligent student of philosophy, acquainted with music, not ignorant of medicine, learned in the responses of jurisconsults, and familiar with astronomy and astronomical calculations”

– Vitruvius, 25 B.C.
Classical Virtues of Architecture

- **Utilitas** – usefulness; appropriate spatial accommodation

- **Firmitas** – structural soundness; robustness toward change

- **Venustas** – aesthetics and beauty; ability of a form to communicate its function and to fit into its environment

*Ref: Vitruvius, De Architectura.*
What Is an Architect?

A creative and respected mediator between a “client” and a “builder”

- Envisions solutions to the client’s need that are feasible and that satisfy all the client’s requirements
- Communicates the vision to the client in a way that is compelling and leads to a decision to implement
- Communicates the vision to the builder in a way that results in its successful implementation
- Inspects and manages the implementation process to ensure that it is brought to a successful conclusion
What Is a System Architect

A creative and respected mediator between a “client” and a “design team”

– Envisions solutions to the client’s need that are feasible and that satisfy all the client’s requirements

– Communicates the vision to the client in a way that is compelling and leads to a decision to implement

– Communicates the vision to the design team in a way that results in its successful implementation

– Inspects and manages the implementation process to ensure that it is brought to a successful conclusion
The Role of a System Architect

Challenges

– Determine user needs & how value is created
– Effectively manage complexity
– Accommodate change
– Work within organizational constraints
– Balance all of the above

Required Skills

– Technical – To make it work
– Communication – To persuade others
– Political – To gain access to scarce resources necessary for the project’s success
Perspectives of a Classical Architect

- **Objective:** Achieve a good “fit” between the form to be designed and its context
  - Form – that over which we have control
  - Context – that which puts demands on the form
- **Challenge:** “We seek to produce harmony between a form we have not yet designed and a context we cannot fully describe.”
- **Process:**
  - Define the context as a set of requirements
  - Partition the requirements into clusters that are richly connected internally and as independent of each other as possible
  - Design a form that meets each cluster of requirements
  - Integrate the low level forms to produce the desired whole

System Architecture

• **System** – A set of components that work together to accomplish a common purpose

• **System Architecture** – The fundamental structure of a system: its elements, the roles they play, and how they are related to each other and to their environment
What are the key components?
How do they work together?
How else could this be done? Why was it done this way?
• **System** - A set of components that work together to accomplish a common purpose

• **System Architecture** - The fundamental structure of a system: its elements, the roles they play, and how they are related to each other and to their environment

• **System Design** - an instantiation of the system architecture
All architecture is design; not all design is architecture.

<table>
<thead>
<tr>
<th>Scope</th>
<th>Design</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>For a specific implementation</td>
<td>Can guide multiple designs and therefore implementations</td>
</tr>
<tr>
<td>Goals</td>
<td>Ensures the specific implementation meets specified requirements (functional, non-functional, and project)</td>
<td>Ensures a number of implementations meet wider business targets and strategies</td>
</tr>
<tr>
<td>Decision Points</td>
<td>Rationale for decisions is driven by implementation parameters (time to market, cost, features, resourcing, etc.)</td>
<td>Rationale for decisions is driven by business strategies (market responsiveness, competitive positioning, etc.)</td>
</tr>
<tr>
<td>Detail</td>
<td>Defines all elements and interfaces to an implementation level of detail</td>
<td>Defines some elements of the system as abstract types, and some interfaces at an implementation level of detail</td>
</tr>
</tbody>
</table>
System Architecture

• **System** – A set of components that work together to accomplish a common purpose

• **System Architecture** – The fundamental organization of a system embodied in its components, their relationships to each other and to the environment, and the principles guiding its design and evolution. *Ref: ANSI/IEEE 1471-2000*

  – Also, the artifacts that describe this structure
System Architecture -
“The Map Is Not the Territory”

System Architecture

Architectural Products
Traditional Views of a Building Architecture

Plan View

Front Elevation View

Side Elevation View
Today’s architects don’t just create documents, they build **models**.
SE Practices for Describing Systems

**Past**

- Specifications
- Interface requirements
- System design
- Analysis & Trade-off
- Test plans

**Present**

Moving from Document-Based to Model-Based Architecture
Model-Based Architecture

- A system model:
  - Is the primary product of model-based systems engineering
  - Incorporates all the system requirements, functional elements, physical components and the relationships between them in a single repository
  - Requires some sort of tool, since there is no way to represent all this information at a single time
  - Is able to provide consistent “views” required by the system designers and a wide range of stakeholders, all derived from the same source
Traditional Views of System Architecture

• **Operational View**
  – Defines the system from the users’ (or operators’) point of view – how the users will interact with the system and how the system will interact with external systems
  – Shows the flow of data (inputs/outputs) between the system, its users and the external systems.

• **Functional View**
  – Defines what the system must do – the capabilities or services it will provide and the tasks it will perform
  – Shows the messages / data between functions

• **Physical View**
  – Defines the partitioning of the system* resources (hardware and software) needed to perform the functions. Also shows the interconnections between the resources - usually defined by the SE to the configuration item level – Computer Software (CSCI) or Hardware (HWCI)
Operational View
- Shows how the operator will use the system.
- Shows inputs and outputs to users and other systems.
- Usually described by:
  • Operational Concept
  • Context Diagram
  • Use Case Scenarios
  • Sequence Diagrams
  • High Level Data Model

Functional View
- Defines the capabilities, the services, or the functions provided by the system.
- Shows the messages/data between functions.
- Usually described by:
  • IDEF0 Diagrams
  • Functional Flow Block Diagrams
  • N2 Diagrams

Physical View
- Defines allocated resources (hardware and software).
- Shows the interconnections between the resources.
- Usually described by:
  • Physical Block Diagrams
  • Physical Interface Definitions
Part 2: The Functional View
Functional View of a System

- **Elements** – functions; the tasks the system performs

- **Interrelationships** between the elements – the data, information, material or energy exchanged between functions in order to accomplish the tasks
Functions in Systems Engineering

• A function is a *process* that transforms inputs into outputs

• A function describes an *action* taken by the system or by one of its elements

• A function is represented by a *verb* or *verb-noun phrase*
Functional Decomposition

- External Function 1
  - Function Of Interest
    - Sub-function 1
    - Sub-function 2
    - Sub-function 3
    - Sub-function 4
    - Sub-function 5
  - External Function 2

- External Function 1
  - Function Of Interest
    - Sub-function 1
    - Sub-function 2
    - Sub-function 3
    - Sub-function 4
    - Sub-function 5
  - External Function 2
Decomposition vs. Composition

• **Decomposition (top-down)**
  - Partition system function a level at a time
  - Need sound definition of all inputs & outputs

• **Composition (bottom-up)**
  - Define many functionalities (bottom-level functions)
  - Synthesize functional hierarchy from many bottom-level functions

• Use both to develop the best solution
Functional Building Blocks

- Signal functions - generate, receive, transmit, convert or distribute signals
- Information functions - analyze, interpret, query, store or convert information
- Material functions - support, enclose, store, reshape or transport matter
- Energy functions - generate, transform, or control: thrust, torque, power, heat or motion

More examples?

Question - Are these elements hardware or software?
Part 3: ATM Example
ATM Example: Scenario Descriptions

### Sequence Diagram

- **Bank Customer**
  - General ID
  - Request for Unique ID
  - Unique ID
  - Request for Authorization
  - Authorization
  - Request for Transaction

- **ATM System**

- **Bank Computer**

### Activity Diagram

<table>
<thead>
<tr>
<th>Bank Customer</th>
<th>ATM System</th>
<th>Bank Computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter General ID</td>
<td>Request Unique ID</td>
<td>Request Authorization</td>
</tr>
<tr>
<td>Request Unique ID</td>
<td></td>
<td>Request Transaction</td>
</tr>
<tr>
<td>Authorize Customer</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ATM Example: Inputs and Outputs

Sequence Diagram

Bank Customer → ATM System → Bank Computer

- General ID
- Request for Unique ID
- Unique ID
- Request for Authorization
- Authorization
- Request for Transaction

Provide ATM Services

- General ID
- Unique ID
- Authorization
- Request for Unique ID
- Request for Authorization
- Request for Transaction
ATM Example:
First-Level Functional Decomposition

Provide ATM Services
ATM Example:
First-Level Functional Decomposition

Provide ATM Services

Authorize Customer
Communicate with Customer
Determine ATM Responses
Communicate with Bank
ATM Example: Input and Output Assignment

Authorize Customer

Communicate with Customer

Determine ATM Responses

Communicate with Bank

General ID

Unique ID

Authorization

Request for Unique ID

Request for Transaction

Request for Authorization
ATM Example: Sample Functional Flow

1. **Authorize Customer**
   - General ID
   - Transaction Trigger
   - Unique ID
   - Image

2. **Communicate with Customer**
   - Unique ID
   - Transaction Trigger
   - Unique ID

3. **Determine ATM Responses**
   - Authorize Trigger

4. **Communicate with Bank**
   - Request for Unique ID
   - Request for Transaction
   - Request for Authorization

5. **Authorization**
   - Image
### ATM Example: N² Chart - External I/O

<table>
<thead>
<tr>
<th>General ID</th>
<th>Unique ID</th>
<th>Auth</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorize Customer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Communicate with Customer</td>
<td>REQ for UID, REQ for Trans</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Determine ATM responses</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Communicate with Bank</td>
<td>REQ for Auth</td>
</tr>
</tbody>
</table>
### ATM Example: N² CHART - COMPLETE

<table>
<thead>
<tr>
<th>General ID</th>
<th>Unique ID</th>
<th>Auth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorize Customer</td>
<td>UID Trigger/TRANS Trigger</td>
<td>AUTH Trigger</td>
</tr>
<tr>
<td>UID Image</td>
<td>Communicate with customer</td>
<td>Req for UID/Req for TRANS</td>
</tr>
<tr>
<td>Auth Image</td>
<td>Determine ATM Responses</td>
<td>Communicate with bank/Req for Auth</td>
</tr>
</tbody>
</table>
How well does the architecture work?
How do you know?

- Authorize Customer
- Communicate with Customer
- Determine ATM Responses
- Communicate with Bank

Flowchart:
- General ID → Authorize Customer
  - Unique ID Trigger
  - Transaction Trigger
  - Unique ID

- Communicate with Customer
  - Unique ID
  - Image

- Authorization Trigger
  - Request for Unique ID
  - Request for Transaction

- Determine ATM Responses

- Communication Image
  - Authorization Image
  - Request for Authorization
Deciding How the System Will Work

Key Questions:

1. What top-level functions will your system have to perform in order to accomplish it’s mission?
   - **Assessment Criteria:** Top-level functions are few in number (3-6) and fully represent the required system functionality.

2. How else could you have partitioned the system functionality and why did you choose to partition it the way you did?
   - **Assessment Criteria:** Alternative decomposition is plausible and the rationale for selecting the chosen partition is sound.

3. How will the top-level functions interact with each other and with their external environment?
   - **Assessment Criteria:** Key operational scenarios can be traced through the top-level functions.

4. How well will your functional architecture perform and what evidence can you provide to support that assessment?
   - **Assessment Criteria:** The functional architecture has been analyzed to verify that it achieves the key system requirements.