ENSE 622 Systems Engineering Requirements, Design and TradeOff

System-Level Design

Mark Austin

E-mail: austin@isr.umd.edu

Institute for Systems Research, University of Maryland, College Park
Table of Contents

Part 1. Motivation and Approach

- Motivation for System-Level/Architecture-Level Design

Part 2. System-Level Design Procedures

- Behavior-to-Structure Mapping
- Coarse- and Fine-Grained Clusters
- Creating the Detailed (Physical) Design
- Hardware/Software Co-Design
- Working with the Linked Model and Visualization of Behavior

Part 3. Case Study

- Development of a Home Theater System
Part 1. Motivation and Approach
Motivation/Observations

- The world is now highly sensed and interconnected.
  
  We can measure almost anything of importance to a desired level of accuracy.
  
  Business operations can be truly global.

- These advances allow for the design of systems that are highly automated.

- Designers will need to be ...

  ... more productive just to keep the development times and overall complexities of design in check.
Motivation and Approach

Approach

• The pathway forward can be found by looking to the past, where ...
  ... major increases in designer productivity have nearly always been accompanied by new methods for solving problems at higher levels of abstraction.

• From a process perspective,
  ... engineers like to work at high levels of abstraction because it is easier to express the required functionality and constraints.

Long-Term Objective

• Therefore, looking forward we seek ...
  ... new ways of working at higher levels of abstraction,
  and
  ... maximizing opportunities for adaptation by delaying decisions on implementation (e.g., software vs hardware) for as long as possible.
**Example.** Evolution of Abstractions in Software Development

Machine code, assembly language, high-level languages (e.g., Fortran), object-oriented programming (e.g., Java), scripting languages (e.g., Python).
Magic Square of System Development  
(Source: Weiringa, 1998)

The process of design can be viewed as a combination of system decomposition and design refinement processes.

Architecting: selection, positioning, and connection of components.
System-Level View of Design

At the system level, designs are viewed as collections of...

...large, arbitrarily complex functional units, which typically form the major components of a system.

Maximize reuse by having the functional units say nothing about the details of implementation.

Hierarchy and Network Elements

Hierarchy Abstraction

- Module A contains modules B and C

Network Abstraction

- Module D is connected to module E
Part 2. System-Level Design

Part 2. System-Level Design Procedures
Design Procedure ... for a system that does not exist.

Design begins with the development of:

- A functional specification for what the system must do, plus
- Sets of constraints for required performance and cost.

The functional specification and constraints serve complementary roles in the design process:

- The functional specification defines the problem space ... what is the design problem to be solved?
- Constraints define the permissible design space ... what design solutions are acceptable?

Thus, system-level design is concerned with the development of ...

... solution spaces that overlap problem spaces.
Basic Procedure. Pathway of system design development that includes mapping of system behavior fragments onto system structure elements and objects.
What a System-Level Design Provides

Hierarchical Decomposition of System Elements

- A decomposition of system-level concerns into logical components, and their relationships to other components and the external environment.

- These relationships can be expressed through logical interface diagrams, logical scenarios, and control requirements.
What a System-Level Design Provides

A Complete View of System-Level Behavior

- We define input/output flow between logical components to achieve desired capability. The result – a composite view of I/O across multiple scenarios.

- When these conditions do not apply, engineers build models of behavior tailored to specific engineering needs (e.g., thermal analysis; circuit analysis; rigid body analysis), and rely on experience to integrate the results of disparate disciplines.

Architectural and Temporal Constraints

- Architectural constraints are defined by relationships – connectivity, adjacency and containment – among the system elements.

- Temporal constraints are defined by relationships among events in scenarios of system behavior (e.g., required sequencing of activities for proper behavior to occur).

Abstraction

- An abstraction of a set of possible physical system implementations, which can be dealt with later on.
The mapping procedure involves three elements:

**Allocation**

- Allocation is the process of assigning requirements/functionality to an answer.
- In practical terms, this boils down to the selections of functional modules and components from a component library.
- The allocation pathway will be most straightforward when each major aspect of system behavior can be handled (independently) by a single module/component in the system architecture.

That is:

```
Tree of functions               mapping                Architectural
in the system                   ------------->          solution
behavior domain.               domain.
```
**Scheduling**

- Scheduling assigns operations in the behavioral description into control steps for the activation and synchronization of functions/functionality.
- The number of functional units needed within any slice of time (or time step) directly corresponds to the number of operations scheduled into it.

**Binding**

- Binding assigns operations to functional units.
- Tasks may include:
  - Logical implementation of system control requirements.
  - Specification of control requirements for each logical component, based on logical scenarios.

After scheduling is complete, the allocation and binding tasks define ...

... flows of data and information through the system architecture in response to external events.
Example 1. Trivial Mapping

Task/component interaction matrix for system behavior mapped onto structure.

Model of System Behavior

Mode of System Structure

Task / Component Interaction Matrix.

Layout and connectivity of components.
Example 1. Trivial Mapping

This simple task-to-component mapping can be expressed:

<table>
<thead>
<tr>
<th>Task</th>
<th>(is mapped to)</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task A</td>
<td>...</td>
<td>A</td>
</tr>
<tr>
<td>Task B</td>
<td>...</td>
<td>B</td>
</tr>
<tr>
<td>Task C</td>
<td>...</td>
<td>C</td>
</tr>
</tbody>
</table>

Remark. At this point in the design, ...

...the precise purpose of the connectivity is not eminently important.

An electrical system connection might carry a current, a computer system it might carry data or information, a mechanical system connection may act as a pathway for transferring a force to a supporting foundation.
Coarse- and Fine-Grained Clusters

The degree of coupling between tasks is an important metric for final design implementations because ...

... the higher its value, the higher the cost of communication among modules.

Finding the right balance in criteria is an art:

1. If the number of clusters is too small, then the mapping has few chances to attain a good hardware/software trade-off.

2. If the number of clusters is too high, then the problem of mapping is too complex.
Example 2-1. Mapping to Coarse- and Fine-Grained Clusters

Coarse mapping of system behavior onto system structure...

Configuration 1

This simple architecture is partitioned into two subsystems, and thus has few interfaces. As a result, there is limited downstream opportunity for trade-off in selection of hardware and software.
Example 2-2. Mapping to Coarse- and Fine-Grained Clusters

Fine mapping of system behavior onto system structure...

Configuration 2

More complicated architecture; more interfaces; better downstream opportunity for trade-off in selection of hardware and software.

Also notice the use of a facade interface to simplify connectivity between the two main subsystems.
Examples of Behavior-to-Structure Mapping

Adding Control to the Behavior-to-Structure Mapping

A few observations:

- The functional hierarchy viewpoint says nothing about inputs and outputs.
- Connectivity and ordering of functions is handled by a second view that also incorporates control of flow.

We need to know:

- What happens and in what order?
- What are the corresponding inputs and outputs?
Example 3. Adding Control to the Behavior-to-Structure Mapping

System behavior defined through decomposition and ordering of functions....
General Guidelines

1. Try to design components that will serve a single function (high cohesion).
2. Try to map individual elements of system behavior/functionality onto individual elements of system structure.
3. Promote component isolation (i.e., through encapsulation and information hiding);
4. Promote common functions, inputs and outputs;
5. Promote commercial off-the-shelf (COTS) reuse.

Observations:
Items 1 and 2 lead to simplified maintenance strategies.
Item 3 helps to keep the complexity of development in check.
Item 4 simplifies communication among subsystems and, thereby, enhances the opportunity for system expansion.
Item 5 improves the economics of system development through reuse.
Basic Procedure

The physical design shows how the system will be implemented, including tentative choices of hardware and software technology.
Creating the Detailed (Physical) Design

Key Tasks

The key tasks are as follows:

1. Allocate each logical component to one or more allocated components in the architecture framework.
   
   **Note:** Allocated components inherit logical component requirements;

2. Identify candidate technologies and standards.
   
   Allocate logical component functionality to software, hardware and/or manual procedures.

3. Allocate persistent data stores to data components;

4. Identify hardware components to execute software components and store persistent data components;

5. Map system design constraints to allocated components;
Hardware/software co-design attempts to ...

... integrate hardware and software design techniques into a single framework.
Parsing Requirements

In moving from the requirements to models of the system structure and behavior, we need to parse each requirement and identify:

Structure ....

1. The objects that need to be supported in the system structure;
2. The attributes these objects must have;

Behavior ....

1. Fragments of functionally that need to be supported by the system behavior;
2. Metrics for measuring system behavior (e.g., via time, performance, sequence).
Attributes and Functions Associated with each Class

The linked model will have classes containing both attributes and functions, i.e.,

<table>
<thead>
<tr>
<th>Name: of Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute: Variables (logical and numerical)</td>
</tr>
<tr>
<td>Function: Provide a linkage between structure and fragments of behavior.</td>
</tr>
</tbody>
</table>

Tabular Format for Representing Fragments of System Functionality

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Structure</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Objects, Attributes</td>
<td>Time, Performance, Sequence</td>
</tr>
<tr>
<td>Req. 1.0</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>Req. 2.0</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>
**Example.** Design of a Sports car ....

Sample requirements and associated structure and behavior.

<table>
<thead>
<tr>
<th>Sports Car Requirement</th>
<th>Structure</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Req. 1.0.</strong> The car must be comfortable.</td>
<td>Seat, seat material.</td>
<td>measureOfComfort()</td>
</tr>
<tr>
<td><strong>Req. 2.0.</strong> The car must have good acceleration</td>
<td>Car, peakAcceleration.</td>
<td>peakAcceleration()</td>
</tr>
</tbody>
</table>

With these mappings in place we can:

- Write specifications as design constraints.
- Formulate problems for multi-objective tradeoff analysis.
Specification of Design Margins

The range of design flexibility should have ...

... wide margins early in design and tighter margins later.

**Example.** Design Spec for LCD, Combat Marking System (Marsh, 2003)

<table>
<thead>
<tr>
<th>Object</th>
<th>Specifications</th>
<th>No.</th>
<th>Category</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hand Control Unit</td>
<td></td>
<td>1</td>
<td>Component life (uses)</td>
<td>100,000</td>
<td>1,000,000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Weight (g) =</td>
<td>200</td>
<td>1,000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>Cost/unig ($ =</td>
<td>$200</td>
<td>$500</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>Screen area (sq in) =</td>
<td>9.00 sqin</td>
<td>25.00 sqin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>Power usage (W) =</td>
<td>2.0 W</td>
<td>5.0 W</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>Character height (mrad) =</td>
<td>5.5</td>
<td>8.0</td>
</tr>
</tbody>
</table>
Visualization of Behavior

Overview

Visualization of behavior in the linked model is possible with a number of diagramming notations:

1. Sequence diagrams annotated with system states between message passing;
2. Pathways (or traces) of message passing through system structure class diagrams;
3. Pathways (or traces) of message passing through a composite-structure (i.e., port) diagram.
Tagging Scenarios with Composite States

The relationship between scenario and component state is many-to-one, ... meaning that several scenario states can refer to the same component state.

Example
Participants: Vimal Mayank, Natasha Kositsyna, Mark Austin.
Industry: Tom Phillips, Dave Everett at NASA Goddard
Duration: May 2003 – August 2003.

In an effort to better understand how requirements should be written and organized for team-development, we developed a set of requirements for a home theater system.

Statement of Need

My wishes are very modest:

1. I simply want to watch movies on a large size theater screen that is connected to a high fidelity audio system.

Initial Requirements

1. I need a home theater system.
2. The total cost must be less than USD $8,000.
Case Study: Home Theatre System

Flowdown of Requirements to a Detailed System Architecture

Requirements

Home Theatre System
--- I need a home theatre system.
--- The total cost must be less than US $8,000

Speakers
--- Cost < US $600

Amplifier
--- Cost < US $400

Flat Screen TV
--- Cost < US $7,000

Description of System Architecture

--- Collection of system components
--- Symbolic description of component connectivity.
--- Component–level requirements and constraints.
--- Interface requirements.
Organization of Requirements

Requirements are organized into three layers:

1. Initial customer requirements (reflecting the statement of need);
2. Detailed agreement between the customer and builder; and
3. Component requirements.

Level 1. Initial Customer Requirements

**R1** I need a home theater system.

**R2** The total cost must be less than or equal to $8,000.
Level 2. Detailed Agreement between Customer and Supplier

Visual Display

R1 The theater system shall have a large display screen.
R2 The display should enable picture clarity from a wide range of viewing angles.
R3 The display must be thin enough to be mounted on a wall.
R4 Cost of the visual display shall not exceed US $7,000.

Audio System

R1 The system shall have a high fidelity audio system.
R2 Cost of the "audio system" shall not exceed US $1,000.
Implicit Requirements

**R1** All components must be "reliable" commercial off-the-shelf (COTS).

**R2** All components must run off a standard A/C power supply in the house.

Points to note:

1. Overall cost of the system ($8,000) is "apportioned" to allowable budgets for the visual display and audio systems.

2. Implicit requirements emanate from the tight budgetary constraints – if we only have $8,000, then custom-building a home theater system is financially infeasible.
Level 3. Component-Level Requirements

The component requirements need to contain quantitative elements that can constrain the selection of components from a database.

**Flatscreen TV**

- **R1** Geometry of the screen shall be at least 3 ft by 4 ft.
- **R2** The screen thickness shall be no more than 6 inches.
- **R3** Weight of the screen shall be no more than 100 lbs.
- **R4** Mean time to failure for the TV screen shall be at least 100,000 hrs.
- **R5** Cost of the flatscreen TV system $\leq$ US $7000.00
- **R6** Interface requirements – to be determined (TBD).
## Level 3. Component-Level Requirements (cont’d)

### Amplifier System

R1 The price of the amplifier system $< US$400.00 $

R2 The audio system output shall be at least 200 watts, but no more than 300 watts.

R3 Interface requirements – to be determined (TBD).

### Speaker System

R1 Cost of the speaker system $< US$600.00$

R2 Capacity of the speaker system output shall be at least 350 watts.

R3 Interface requirements – to be determined (TBD).
Generation of Requirements-Specification

Requirement-specifications for the home theatre system are generated by ...

... identifying the "quantitative" element for each Level 3 requirement.

The flatscreen TV objects need to have the following attributes:

Screen height, screen width, thickness, weight,

and the speakers will have the performance metric

Output (watts).
## Design Margins for Flatscreen TV

<table>
<thead>
<tr>
<th>Object</th>
<th>Specifications</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>Category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Component life (hrs)</td>
<td>100,000</td>
<td>1,000,000</td>
</tr>
<tr>
<td>2</td>
<td>Weight(lbs)</td>
<td>30</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Cost/unit ($)</td>
<td>$6,000</td>
<td>$7,000</td>
</tr>
<tr>
<td>4</td>
<td>Screen height (ft)</td>
<td>3.00 ft</td>
<td>5.00 ft</td>
</tr>
<tr>
<td>5</td>
<td>Screen width (ft)</td>
<td>4.00 ft</td>
<td>6.00 ft</td>
</tr>
<tr>
<td>6</td>
<td>Thickness (in)</td>
<td>1.0</td>
<td>6.0</td>
</tr>
</tbody>
</table>
Case Study: Home Theater System

Selection of Components Choosing the Amplifier

- I need a home theatre system.
- The total cost must be less than US $8,000

System Design / Architecture

Interface requirements.

"Buy vs Build" subject to constraints.

--- $ Cost of Speakers + $ Cost of Amplifier + $ Cost of TV < US $ 8,000

Library of "Product" Descriptions

---
Selection of Components Choosing the Speakers

**Home Theatre System**
- I need a home theatre system.
- The total cost must be less than US $8,000

**System Architecture**

- **Speakers**
  - Target: Cost < US $600

- **Amplifier**
  - Actual: Cost = US $450

- **Flat Screen TV**
  - Target: Cost < US $7,000

**Trade Space**

- **Interface**
- **Performance**
- **Cost**

"Buy vs Build?" subject to a constrained design space.