Review of ENSE 621/ENPM 641: Part 01

*Introduction to Systems Engineering*

Mark Austin

E-mail: austin@isr.umd.edu

Institute for Systems Research, University of Maryland, College Park
Quick Review of ENSE 621/ENPM 641

Topics:

2. Our definition of Systems Engineering.
5. Models of Systems Engineering Development (e.g., Waterfall, Spiral).
6. Economics of development.
7. Systems Engineering Drivers
8. Strategies for Systems Engineering Development
SYSTEMS ENGINEER: “BEST JOB IN AMERICA”

Money Magazine
Best Jobs in America by Donna Rosato with Beth Braverman and Alexis Jeffries. Oct. 9, 2009
Source: MoneyOnCNNMoney.com

“Money and PayScale.com, a leading online provider of employee-compensation data, surveyed 35,000 people online about what makes a great job, they rated intellectual challenge, a passion for the work, and flexibility just as highly as security.

1. Systems Engineer
   Median salary (experienced): $87,100
   Top pay: $130,000
   Job growth (10-year forecast): 45%
   Sector: Information Technology

What they do: They're the "big think" managers on large, complex projects, from major transportation networks to military defense programs. They figure out the technical specifications required and coordinate the efforts of lower-level engineers working on specific aspects of the project.

Why it's great: Demand is soaring for systems engineers, as what was once a niche job in the aerospace and defense industries becomes commonplace among a diverse and expanding universe of employers, from medical device makers to corporations like Xerox and BMW. Pay can easily hit six figures for top performers, and there's ample opportunity for advancement. But many systems engineers say they most enjoy the creative aspects of the job and seeing projects come to life. "The transit system I work on really makes a tangible difference to people," says Anne O’Neil, chief systems engineer for the New York City Transit Authority.

Drawbacks: Long hours are common; project deadlines can be fierce.
Pre-reqs: An undergrad engineering degree; some jobs might also require certification as a certified systems engineering professional (CSEP)."

Systems engineering is a discipline that lies at the cross-roads of engineering and business concerns.

Specific goals are to provide:

1. A balanced and disciplined approach to the total integration of the system building blocks with the surrounding environment.

2. A methodology for systems development that focussed on objectives, measurement, and accomplishment.

3. A systematic means to acquire information, and sort out and identify areas for trade-offs in cost, performance, quality etc....
Typical concerns on the **design side**:

1. What is the required functionality?
2. How well should the system perform?
3. What about cost/economics?
4. How will functionality/performance be verified and validated?

Typical concerns on the **management side**:

1. What processes need to be in place to manage the development?
2. What kind of support for requirements management will be needed?

**Learning how to deal with these concerns in a systematic way is a challenging proposition driven, in part, by a constant desire to improve system performance and extend system functionality.**
Emergence of Information-Centric Systems

Nowadays the results of this trend can be found in the...

... replacement of industrial-age systems (primarily hardware) with information-centric systems (hardware, software, and communications), so-called because the teams of people responsible for the design, manufacture, and operation of these systems are likely to be geographically dispersed and, therefore, ...

... rely on flows of data and information for the coordination and control of development activities and system control.

These new systems have ...

... extended functionality and better performance (compared to their predecessors), but the design task is now much more difficult than before.
Understanding a System

To understand a system you really need to figure out:

1. The ways in which it will be used,
2. The environment in which it will operate, and
3. The knowledge, technologies, and methods that go into making it.

Handling System Complexity

As systems become more complex, ...

... we need to be strategic in the way we approach design.

This points to the importance of:

1. **Abstractions** (to simplify decision making in design).
2. **System Decomposition** (to simplify design).
3. **Formal Analysis** (our understanding of system behavior needs to be right).
**Strategy:** Put original problem aside and focus on understanding the collection of subsystems that make up the original system.

Common questions include:

1. What are the subsystems and how are they connected internally?
2. How does the system interact with the surrounding environment?
System Assembly via Integration of Abstractions

System assembly through integration of abstractions

Complex System

Subsystem

Components

Observations

Increasing importance of technology

Increasing range of functionality

Increasing opportunity for reuse of lower level entities

Engineering Concerns

Increasingly heterogeneous

Increasingly homogeneous

Increasing use of abstraction

Increasing need for formal analysis
Modern buildings are:

... advanced, self-contained and tightly controlled environments designed to provide services (e.g., transportation, artificial lighting, ..etc.).

The design of modern buildings is complicated by:

1. Necessity of performance-based design and real-time management.

2. Many stakeholders (owners, inhabitants), some with competing needs.

3. Large size (e.g., 30,000 occupants; thousands of points of sensing and controls for air quality and fire protection.)

4. Intertwined network structures for the arrangement of spaces, fixed circulatory systems (power, hvac, plumbing), dynamic circulatory systems (flows of energy through rooms; flows of material).
Framework for interaction of architectural, structural, control, and networked embedded system design activities.
## Case Study: SE for Modern Buildings

<table>
<thead>
<tr>
<th>System Level</th>
<th>Subsystem Level</th>
<th>Component Level</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Architectural Concerns</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Form and functionality. Services, access, comfort.</td>
<td>Floor level spaces, positioning of spaces, connectivity among spaces.</td>
<td>Walls and spaces, portals, doorways, windows ...</td>
</tr>
<tr>
<td><strong>Structural Concerns</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structural assemblies, overall system stability</td>
<td>Frame, floor, and wall systems. Forces, deflections.</td>
<td>Beam and column elements, beam/column joints, material behavior.</td>
</tr>
<tr>
<td><strong>Electro-mechanical Concerns</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access, comfort, safety</td>
<td>HVAC, lighting, fire protection</td>
<td>Heat exchangers, pipes, elevators, escalators, sprinklers</td>
</tr>
</tbody>
</table>
Traditional approaches to building automation design follow a top-down process with very little feedback.

Factors Driving Design
- Architectural constraints.
- Occupancy requirements.
- External loads (gravity, thermal).
- Control requirements.
- Comfort requirements.
- Control speed requirements.
- Sensor and actuator requirements.
- Layout requirements.

Design Flow
- Architecture Design
- Control Design
- Network Design
- Implementation

Flowdown of constraints
- Layout constraints.
- Sequence of operations.
- Actual implementation cost.
Case Study: SE for Modern Buildings

Design Flow for Platform-Based Design

Factors Driving Design

- Architectural requirements.
- Occupancy requirements.
- External loads (gravity, thermal, ...)
- Ventilation requirements.
- Energy generation requirements.
- Sequence of operations.
- Comfort requirements.
- Control speed requirements.
- Sensor and actuator requirements.
- Layout requirements.

Design Flow

1. Architecture Design
2. Power Design
3. Control Design
4. Network Design
5. Implementation

Performance

- Maximum ventilation.
- Maximum power generation.
- Cost estimates.
- Minimum response time.
- Control accuracy.
- Maximum available bandwidth.
- Maximum computational speed.
- Maximum storage size.
- Actual ventilation.
- Actual power generation.
- Actual network speed.
- Actual layout constraints.
- Actual installation cost.
Two Examples of Building-Integrated Energy Systems Design

The left-hand collage shows the Pearl River Complex currently under construction in Guangzhou, China.

The right-hand collage illustrates the use of wind and solar power mechanisms, and vacuum driven ventilation in the Chicago-bound Clean Technology Tower.
Traditional engineering and systems engineering serve complimentary roles:

- **Traditional Engineering.**
  
  Focus on generation of knowledge needed to create new technologies and new things.

- **Systems Engineering.**
  
  Focus on understanding how existing technologies and things can be integrated together in new ways (...to create new kinds of systems).

So here’s the bottom line:

... systems engineers need traditional engineers, and vice versa.
Focus on:

...liaison among disciplines, supported by formal methods for systems analysis and design.
Systems are developed by teams of engineers – the team members must be able to understand one-another’s work.
SE at the Project Level

Key concerns:

1. Synthesis of requirements that might extend beyond functionality, performance and cost (e.g., social concerns, political concerns, long-term sustainability).

2. Partitioning of the design problem into several levels of abstraction and viewpoints suitable for concurrent development by design teams;

3. Synthesis of good design alternatives from modular components;

4. Integration of the design team efforts into a working system; and

5. Evaluation mechanisms that provide a designer with critical feedback on the feasibility of a system architecture, and make suggestions for design concept enhancement.

6. Formal methods for early validation/verification of systems.
SE at the Product Level

Use of available technology to achieve overall best cost–benefit result.

Features
- Functionality
- Cost
- Reliability
- Interfaces

Representation

New Knowledge
- New Technologies
- New Things
Key concerns:

1. How to describe what a product does? Can this be done formally?
2. How to describe pre-conditions for using a product?
3. How to describe a product's interfaces?
4. How to describe various representations (visual, mathematical).
The principal products of systems engineering development are as follows:

- Requirements specification; system (logical) architecture; system (physical) design; the physical system itself.

These products are produced by the following processes:

- Requirements engineering; system architecting; systems design and integration; optimization and trade-off analysis; validation and verification.
Technical process for system architecting.....

End-to-End Lifecycle Development

Needs → Requirements Analysis → Function Analysis → System Synthesis

Assessment of Risks and Uncertainty

Control Factors
-- Requirements
-- Functions
-- Reuse of components

System Architecture
-- Cost estimate
-- Performance estimate
-- Schedule
The terms system validation and verification refer to two basic concerns, “are we building the right product” and “are we building the product right?” Satisfactory answers to both questions are a prerequisite to customer acceptance.

Validation and verification concerns are a prerequisite to customer acceptance.
Pre-defined plans of development ...

... provide the discipline to keep development activities predictable and on track.

The project participants know what’s expected and when.

**Interaction of technical development and engineering management processes**

During the past 3-4 decades this approach to system development has served many industry sectors (e.g., aerospace) well.
## Systems Engineering Processes

### Engineering/Systems Engineering Activities and Artifacts

<table>
<thead>
<tr>
<th>Engineering Activity</th>
<th>Systems Engineering Activity</th>
<th>Artifact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Analysis</td>
<td>Requirements Analysis</td>
<td>Requirements baseline and specification.</td>
</tr>
<tr>
<td>Architecture</td>
<td>Function/behavior analysis</td>
<td>Logical Architecture.</td>
</tr>
<tr>
<td>Design</td>
<td>Synthesis</td>
<td>Physical Architecture and Design</td>
</tr>
</tbody>
</table>
# Systems Engineering Management Activities and Artifacts

<table>
<thead>
<tr>
<th>Management Activity</th>
<th>Systems Management</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Management</td>
<td>Activity</td>
<td>Artifact</td>
</tr>
<tr>
<td></td>
<td>Requirements Manage-ment</td>
<td>Requirements baseline and specification.</td>
</tr>
<tr>
<td>Configuration Management</td>
<td>Activity</td>
<td>Artifact</td>
</tr>
<tr>
<td></td>
<td>Planning of activities and tasks. Communicate compliance status</td>
<td>Work products</td>
</tr>
<tr>
<td>Baseline Management</td>
<td>Activity</td>
<td>Artifact</td>
</tr>
<tr>
<td></td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
</tbody>
</table>
The waterfall model works well when:

... problem and solution method are well understood, requiring no large-loop corrections to development problems.
Limitations of Waterfall Model

- Changing requirements proved to be the biggest cause of cost overruns and schedule slips in the waterfall era.

- Users were found to be unable to define the requirements of a complex system without having had hands-on previous experience with the system – A Catch 22.
Spiral Model of Systems Development

Spiral model corresponds to risk oriented iterative enhancement.

Categories of risk include: technical risk, schedule risk, cost risk, programmatic risk.
Flowdown of requirements in the V-Model of system development.

- Stakeholder Requirements
- System Requirements
  - Subsystem Requirements
    - Component Requirements
      - Component Design
      - Component Test
  - Subsystem–Level Design
    - System–Level Design
      - System Requirements
        - System Test
      - Validate the system
      - Verify the system
      - Validate the system
      - Validate the system
- Stakeholder Test

Allocate requirements to components.

Design Problem Definition

Implementation and Test
Functional flow block diagram for the core technical process at GE.

- Assess Available Information
- Create Behavior Model
- Evaluate Effectiveness Measures
- Create Structure Model
- Check design for defects
- Improve defects at lower level
- Perform Trade-off Analysis
- Create Sequential Build – and Test Plan
- Reduce defects via reallocation of resources
- Iterate to find feasible solution
Funding Commitments in Product Life-Cycle

- Preliminary Design
- Commence Production
- Funds Committed
- Funds Expended

Cumulative Percentage vs. Product Lifecycle
Knowledge Gap in Systems Development

- **Cumulative Percentage**
- **Product Lifecycle**

- **Funds Committed**
- **Knowledge**
- **Funds Expended**
- **Ease of change**

- **Preliminary Design**
- **Commence Production**

- Knowledge Gap
## Cost of Correcting Design Errors

<table>
<thead>
<tr>
<th>Project Phase</th>
<th>Bug Description</th>
<th>Relative Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>Design Team</td>
<td>1</td>
</tr>
<tr>
<td>Write and Test</td>
<td>Designer</td>
<td>10-20</td>
</tr>
<tr>
<td>Quality Assurance</td>
<td>QA Personnel</td>
<td>70-100</td>
</tr>
<tr>
<td>Shipment to Customer</td>
<td>Customer</td>
<td>Very-expensive</td>
</tr>
</tbody>
</table>
The key message is as follows:

... look at where the costs are, and then focus on parts where savings can be made.