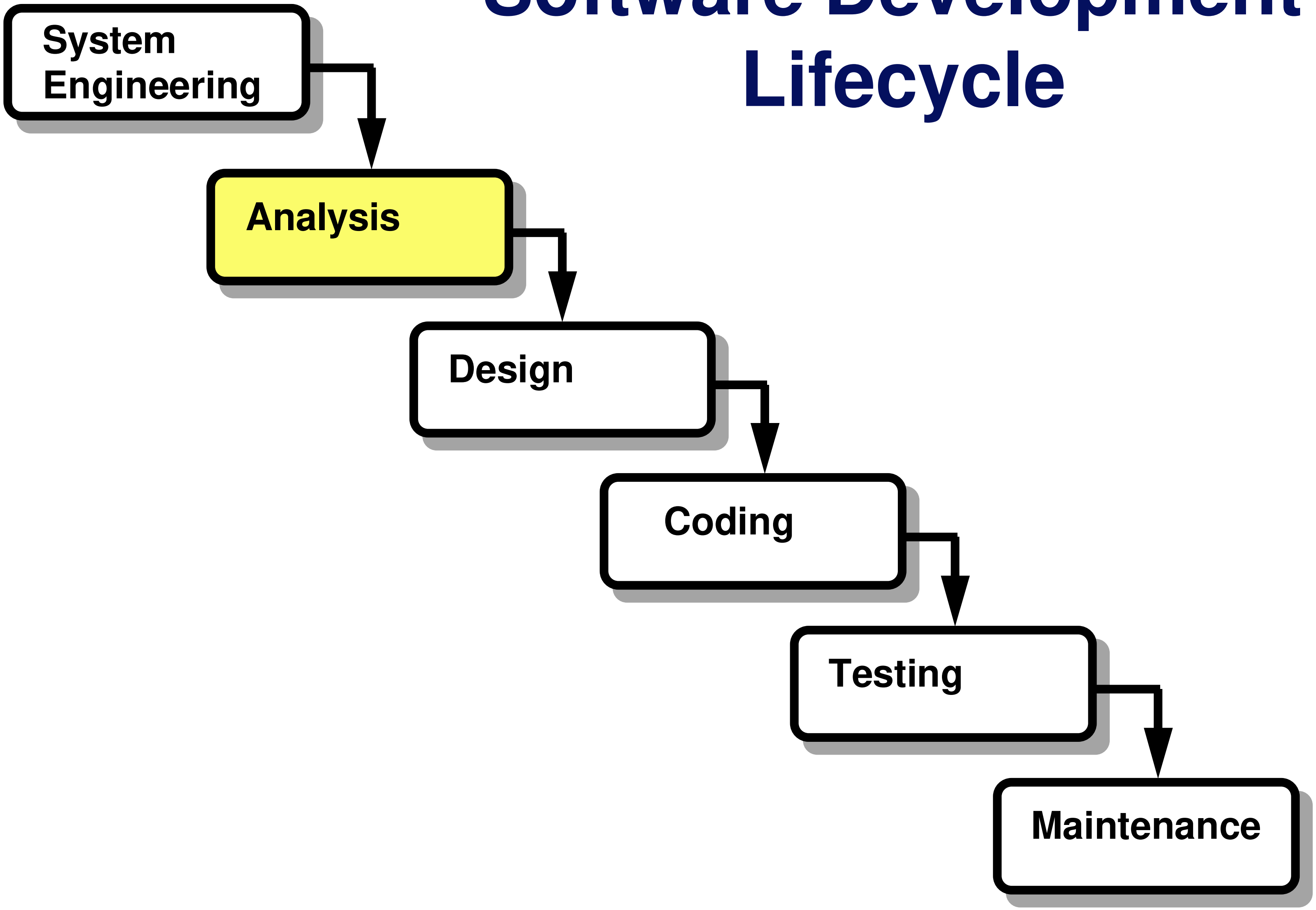
TOPICS

Fundamentals

Structured and Object-Oriented Analysis

Formal and Automated Techniques

Software Development Lifecycle



System Engineering

Analysis

Design

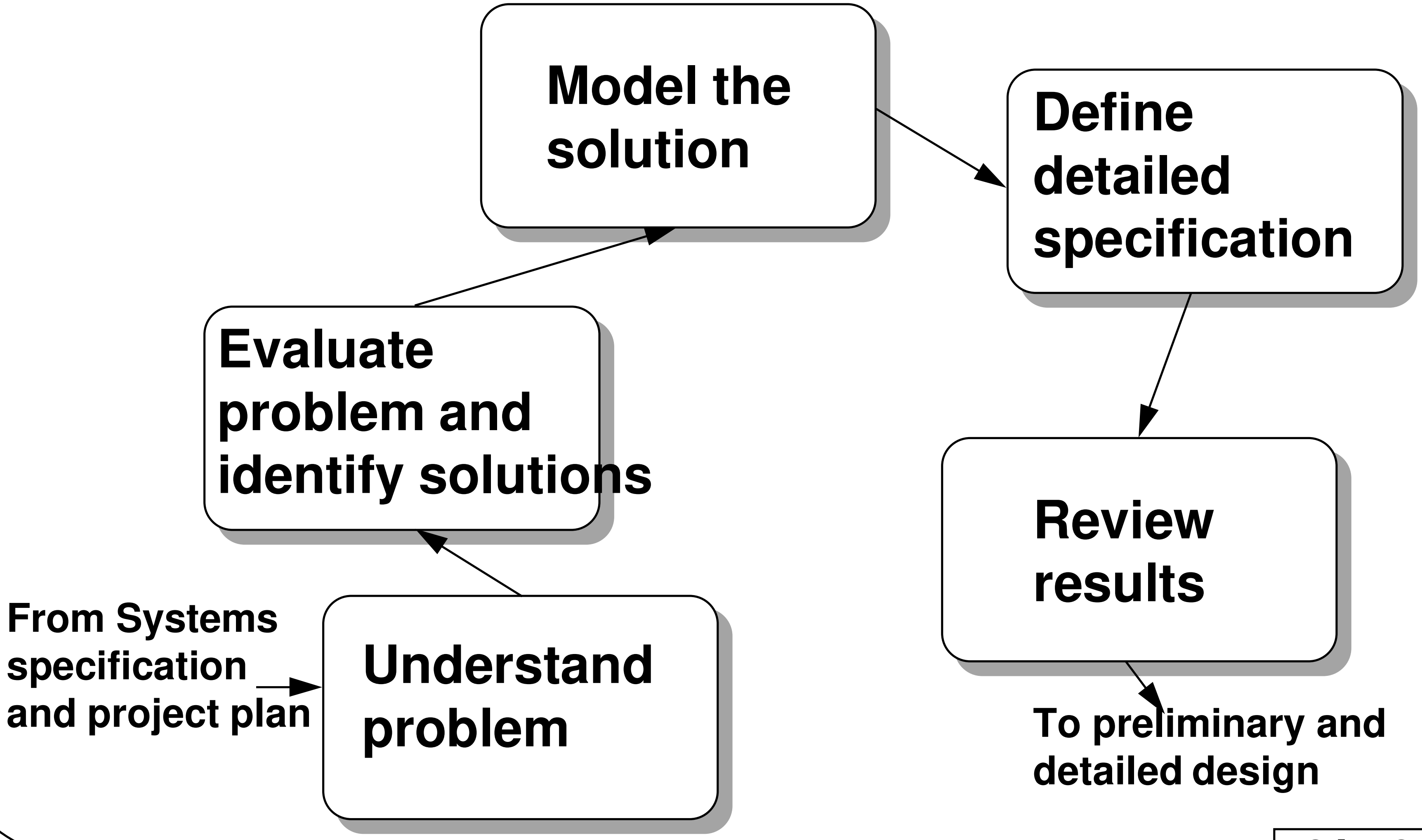
Coding

Testing

Maintenance

Requirements Analysis - Overview

Tasks



From Systems specification and project plan

Understand problem

Evaluate problem and identify solutions

Model the solution

Define detailed specification

Review results

To preliminary and detailed design

Basic Activities of Software Requirements Analysis

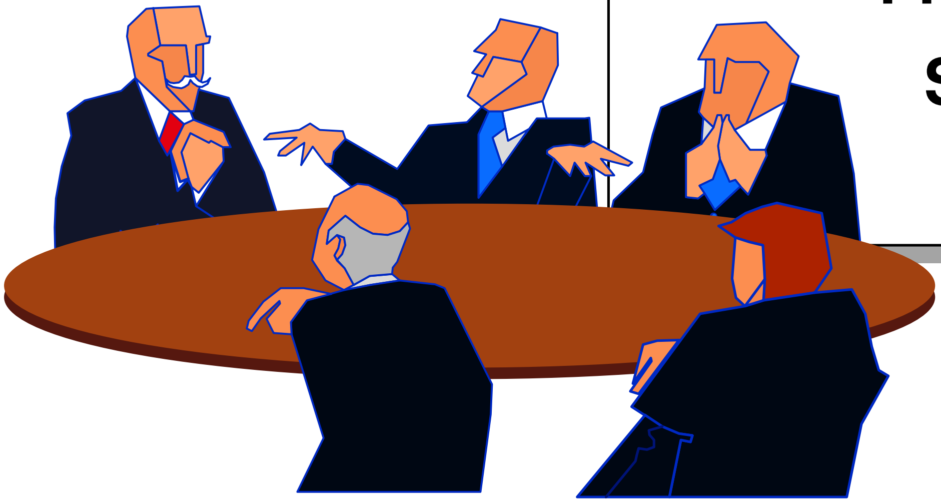
- **Define the functional domain - what functions are to be performed?**
- **Define the information domain - what is the flow of information in the system, what is the structure of that information, and what is the content of that information?**
- **Partition the problem - what is the hierarchy of the problem?**
- **Develop the logical view of the requirements - detail the functions and data**
- **Develop the physical view of the requirements - detail the real-world forms of the functions and data**

Common Problems Encountered During Requirements Analysis

- general communications problems, including not understanding the problem, misinterpreting information, and missing information
- acquiring pertinent information
- handling problem complexity
- accommodating changes that will occur during and after analysis

Beginning the Process

Hold a meeting!



The Facilitated

Application

Specification

Technique (FAST)

Example: The SafeHome System

A microprocessor-based home security system that protects against a number of undesireable events such as illegal entry, fire, flood, etc.

SafeHome will use sensors to detect each situation, can be programmed by the homeowner.

SafeHome will automatically telephone a monitoring agency when a situation is detected.

Problem Understanding

Step 1. Identify objects, operations, constraints, and performance criteria:

Objects

- Smoke detectors
- Window/door sensors
- Motion detectors
- Alarm
- Control panel
- Telephone numbers

Operations

- Set/reset alarm
- Monitor sensors
- Dial phone
- Program control panel

Constraints

- Cost less than $200
- Easy to use
- Direct dial to telephone

Performance Criteria

- Display within 1 s of event
- Prioritize event processing
- Delay at least 1 min before dialing phone

Problem Understanding, Continued

Step 2. Develop "mini"-specification for each entry on each list

Object: <u>Control Panel</u>

Mounted on wall

Size 9x5 inches

Contains 12 key-pad and special keys

Diagram of panel

All user interaction through control panel

Used to enable and disable system

Software to provide interaction guidance, echo responses, etc.

Connected to all sensors

Problem Understanding, Continued

Step 3. After much debate and list modifications, create list of validation criteria

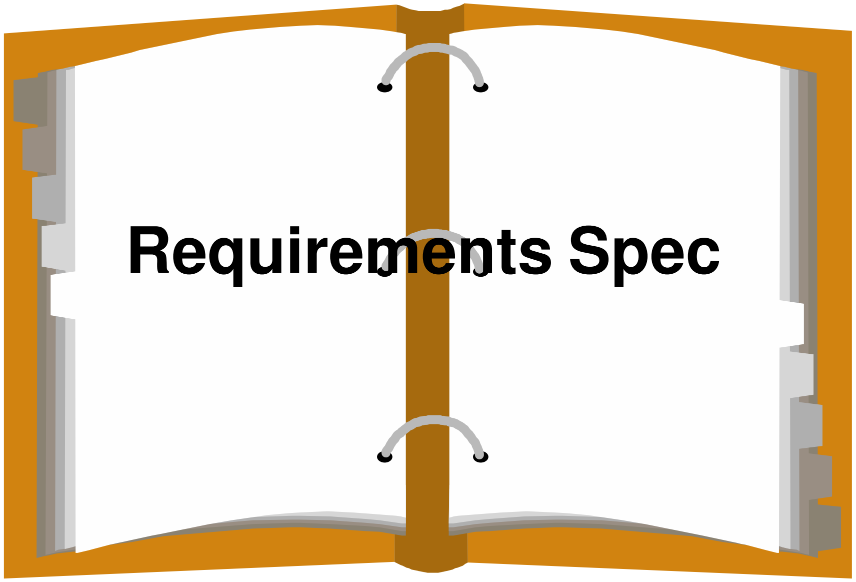
Enter 200 random events and observe alarm responses

Ensure display resets on power up

When phone numbers are entered with 555- prefix, ensure telephone is *not* dialed

Problem Definition

Step 4. Write complete draft specification using results of steps 1-3



Concepts of Analysis

Information Domain:

1. Information flow
2. Information content
3. Information structure

Modeling: Pictorial representation of problem solution

Aids analyst in understanding problem

Focal point of review

Foundation for design

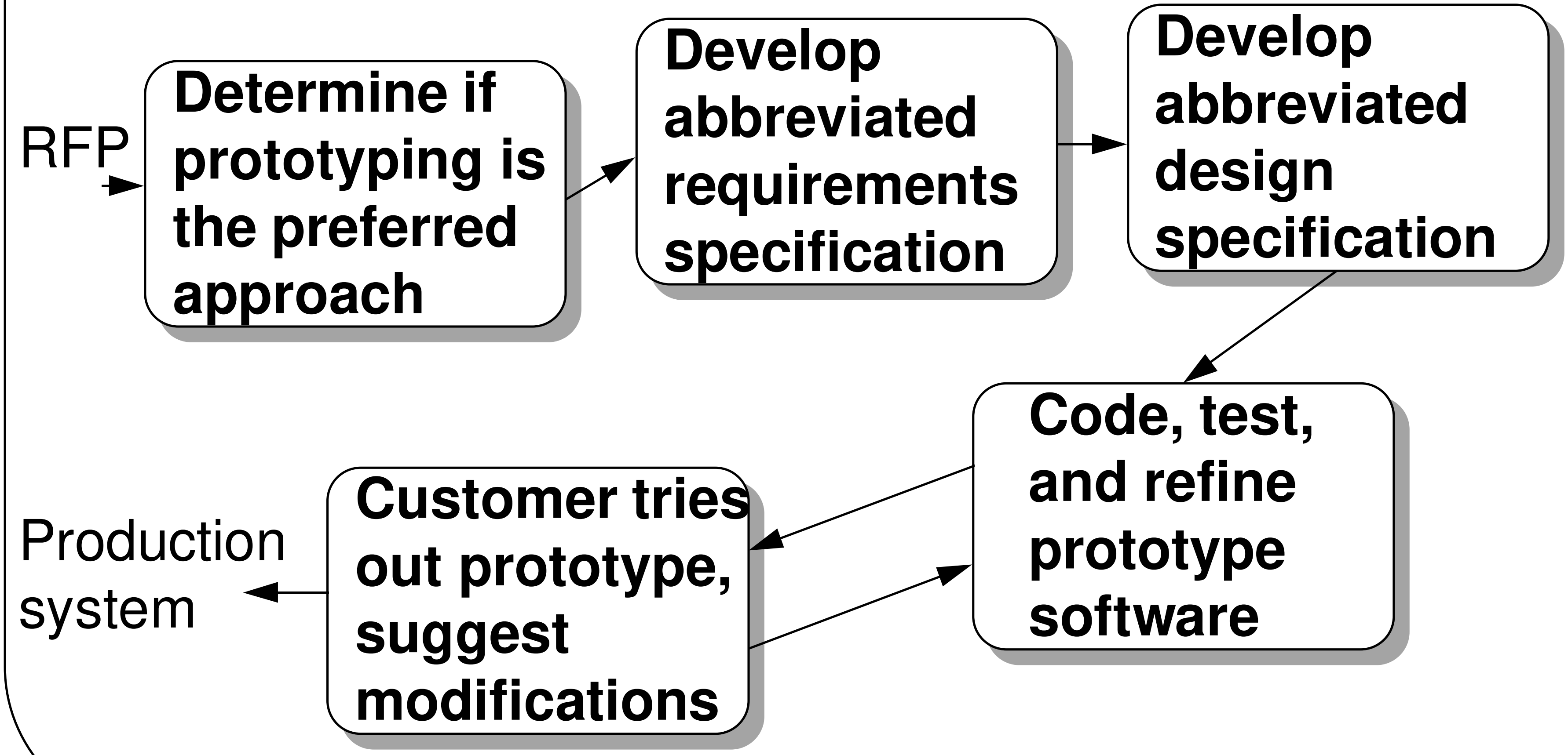
Partitioning: Break big problems into little ones

Software Views

<u>View</u>	<u>Focus</u>
Informational	Data
Functional	Functions
Behavioral	Execution process

Software Prototyping

Assume a request for proposal (RFP) or system spec defines the problem.



RFP

Determine if prototyping is the preferred approach

Develop abbreviated requirements specification

Develop abbreviated design specification

Code, test, and refine prototype software

Customer tries out prototype, suggest modifications

Production system

Specification Principles

- Separate functionality from implementation - describe what is desired, not how
- Understand the system of which the software is a part and the environment in which the system resides
- Develop a cognitive model rather than a design or implementation model, and keep the perspective of the user
- View the specification as a model, see if it is adequate to determine if a proposed implementation is satisfactory, and tolerate imcompleteness
- Localize and loosely couple the specification

Software Requirements Analysis (SRA)

Common Characteristics of the Methodologies

- They perform information domain analysis
- They have a means to represent functions
- They can define interfaces
- They support partitioning of the problem
- They support abstraction
- They can represent both the physical and logical views of the problem