WHAT THE CUSTOMER WANTED

WHAT THE CUSTOMER REQUESTED

WHAT THE CONTRACTOR ORDERED

WHAT ENGINEERING DESIGNED

WHAT MANUFACTURING BUILT

WHAT FIELD SERVICE INSTALLED
Requirements Engineering: Introduction

- Why RE?
  - Why RE in SysE? Software Lifecycle and Error Propagation
  - Case Studies and The Standish Report

- What is RE?
  - Role of Requirements

- How to do RE? -> RE Processes

Sources of Material
Why RE?

Software Lifecycle Revisited

- Systems Engineering
- Requirements Analysis
- Software Architecture and Design
- Implementation
- Testing
- SQA and Metrics
- Maintenance
- Project Planning and Mgmt
- Software Process
- Configuration Mgmt
- Evolution And Re-engineering
Why RE?

Error Propagation in Lifecycle [Mizuno82]

Simplified Lifecycle

Requirements Specification

- correct spec.
- erroneous spec.

Design

- correct design
- erroneous design

Implementation

- correct program
- erroneous program

Testing

- correct functions
- correctable errors
- uncorrectable errors
- hidden errors

Maintenance

Imperfect program products

How big is the erroneous spec.? How costly is it?

Lawrence Chung
Why RE?

How big is the "erroneous specification"?

† Bell Labs and IBM studies
80% of all defects are inserted in the requirements phase. Improving the requirements definition process reduces the amount of testing and rework required.
And the above figures do not include the end user losses who have to live with poor software on a daily basis.[Testing Techniques Newsletter]

† U.S. Air Force projects
36% of all defects were due to faulty requirements translation. Only 9% of these errors were resolved (in the requirements phase).[Sheldon92]

† Voyager and Galileo spacecraft
Of the 197 significant software faults found during integration & system testing, only 3 of those errors were programming errors; the vast majority of the faults were requirements problems.[Lutz93]

† Application Specific Integrated Circuits [ASICs]
>1/2 are faulty on first fabrication. A majority of these faults are related to reqs. errors.

† [UK Health and Safety] Executive
Specification 44.1% Operation and Maintenance 14.7%
Design and Implementation 14.7% Changes after commissioning 20.6%
Installation and Commissioning 5.9%
yearly since 1994, survey of close to 300,000 projects

What Factors Contribute to Project Success?

28% completed on time and on budget
78,000 projects

49% overran original estimates
- Time overrun averaged 63%
- Cost overrun averaged 45%
137,000 projects

23% canceled before completion
65,000 projects

The CHAOS Ten

<table>
<thead>
<tr>
<th>Project Success Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Executive Management Support</td>
</tr>
<tr>
<td>2. User Involvement</td>
</tr>
<tr>
<td>3. Experienced Project Manager</td>
</tr>
<tr>
<td>4. Clear Business Objectives</td>
</tr>
<tr>
<td>5. Minimized Scope</td>
</tr>
<tr>
<td>6. Standard Software Infrastructure</td>
</tr>
<tr>
<td>7. Firm Basic Requirements</td>
</tr>
<tr>
<td>8. Formal Methodology</td>
</tr>
<tr>
<td>9. Reliable Estimates</td>
</tr>
<tr>
<td>10. Other</td>
</tr>
</tbody>
</table>
What Factors Contribute to Project Failure?

The CHAOS Ten

**Issues**

1. Lack of User Input
2. **Incomplete** Requirements & Specifications
3. **Changing** Requirements & Specifications
4. Lack of Executive Support
5. Technology Incompetence
6. Lack of Resources
7. Unrealistic Expectations
8. Unclear Objectives
9. Unrealistic Time Frames
10. New Technology

The CHAOS Ten

**Project Impaired Factors**

1. **Incomplete** Requirements
2. Lack of User Involvement
3. Lack of Resources
4. Unrealistic Expectations
5. Lack of Executive Support
6. **Changing** Requirements & Specifications
7. Lack of Planning
8. Didn't Need It Any Longer
9. Lack of IT Management
10. Technology Illiteracy
“The definition of insanity is doing the same thing over and over again and expecting a different result.”

[Albert Einstein]
Size Is Important: Success by Project Size

Standish Group, ‘99 (www.standishgroup.com)

Success Rate (%)

Project Size ($)

Why?
Why RE?

How costly are requirements errors?

[Lindstrom93]
Get the requirements wrong, you’ll destroy the project.

[Boehm87]

COST (correcting design/implementation errors)
= 100 X COST (correcting requirements errors)

[Humphrey, Managing the Software Process, Ch1, p11-12]

a useful rule of thumb: It takes about 1 to 4 working hours to find and fix a bug through inspections and about 15 to 20 working hours to find and fix a bug in function or system test.

[Curtis88]
Three most frequent problems plaguing large software systems:

communication and coordination
thin spread of domain application knowledge
changing and conflicting requirements

Defining the problem is The Problem

Lawrence Chung
The High Cost of Requirement Errors

Relative cost to repair errors:
When introduced vs. when repaired.

The 1-10-100 Rule

- Requirements Time
  - .5 - 1
  - 2.5
  - 5
  - 10
  - 25
  - 100

- Design
  - Maintenance
- Coding
  - Unit Test
  - Acceptance Test

“All together, the results show as much as a 200:1 cost ratio between finding errors in the requirements and maintenance stages of the software lifecycle.”

Average cost ratio 14:1
[Grady 1989]  [Boehm 1988]

Why?
Why RE?

Productivity  Follows  Quality
[Capers Jones]

- Productivity was the wrong issue.
- Most poor productivity is due to long testing, fixing.
- If get the quality up in the original build, testing goes fast and overall productivity goes up.
- So aim at quality, and you get productivity.
  Aim at productivity rather than quality, get neither.

[Brooks, Jr.]

Why RE?

"The hardest single part of building a software system is deciding precisely what to build.

No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and other software systems.

No part of the work so cripples the resulting system if done wrong.
No other part is more difficult to rectify later."

[Brooks, 1987]
Requirements Engineering: Introduction

- Why RE?
  - Software Lifecycle and Error Propagation
  - Case Studies and The Standish Report

- What is RE?
  - Role of Requirements

- How to do RE? -> RE Processes

Sources of Material
"... Requirements Engineering is the branch of Systems engineering concerned with real-world goals for, services provided by, and constraints on software systems.

Requirements engineering is also concerned with the relationships of these factors to precise specifications of system behavior and to their evolution over time and across system families..." [Zave94]
What is RE?

Role of requirements

Kirchhoff agreement regarding the requirements between system developers, customers, and end-users.

\[\Rightarrow \text{legal contract (flexible, inflexible)}\]
\[\Rightarrow \text{multi-party}\]
\[\Rightarrow \text{communication and coordination}\]
\[\Rightarrow \text{conflicting views}\]
\[\Rightarrow \text{changing views}\]

Kirchhoff the basis for software design

\[\Rightarrow \text{defect-free as much as possible}\]
\[\Rightarrow \text{technically feasible}\]

Kirchhoff support for verification and validation

Kirchhoff support for system evolution

\[\Rightarrow \text{system evolution = change (old system, new system)}\]
\[\text{change (old requirements, new requirements)}\]
Systematic Decision Making is Essential

- Requirements Engineering is about determining
  - problems with the current status (As-Is)
  - objectives to achieve
  - changes to bring about for a better future (To-Be)

We want to make a change in the environment
We will build some system to do it
This system must interact with the environment
What’s Essential?

- Modeling

“A model is a pattern, plan, representation (especially in miniature), or description designed to show the main object or workings of an object, system, or concept” [Wikipedia]

- Systematic decision making

“Decision making can be regarded as an outcome of mental processes (cognitive process) leading to the selection of a course of action among several alternatives. Every decision making process produces a final choice. The output can be an action or an opinion of choice” [Wikipedia]
**What is RE?**

*Not all RE projects are similar*

**Customer-driven projects**
- involve a customer who needs a system that solves a particular problem
- often one-shot

<table>
<thead>
<tr>
<th>Customer</th>
<th>Project</th>
<th>Developer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navy</td>
<td>Missile Tracking</td>
<td>Raytheon-E-Sys</td>
</tr>
<tr>
<td></td>
<td>Weapon Inventory</td>
<td>TRW</td>
</tr>
<tr>
<td>Sprint</td>
<td>MAN</td>
<td>Nortel</td>
</tr>
<tr>
<td>MCI</td>
<td>Accounting</td>
<td></td>
</tr>
</tbody>
</table>

**Market-driven projects**
- involve a developer who needs to develop a system that is to be sold to the market
- often hard to determine what the customer really wants

<table>
<thead>
<tr>
<th>Customer</th>
<th>Project</th>
<th>Developer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Office workers?</td>
<td>Groupware</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Casual users?</td>
<td>Multimedia</td>
<td>Fujitsu</td>
</tr>
</tbody>
</table>

*A Field Study involving 10 organizations [Lubars93]*

**Customer-driven projects**
- usually given large monolithic statements of requirements
- despite their size, these are often sketchy, ill-defined
- concept of "superdesigner" for interpretation, filling gaps
  - needs REeer who can deal with sketchy, ill-defined reqs
  - clarification, completeness through communication

**Market-driven projects**
- often smaller requirements produced in-house
- increasingly important (e.g., from military to commercial, from internal/external customer to open market)

- Securing customer interaction always hard but critical
- About 1/3 of the projects did some sort of prototyping
- Organizations are changing much faster today, for economic and technological reasons
  - Requirements "evolution" a major concern
  - Requirements "traceability" a major concern
  - Requirements must deal with the environment (i.e., EM)

*No such thing as the "problem"*

Lawrence Chung
Requirements Engineering: Introduction

- Why RE?
  - Why RE in SysE?
  - Software Lifecycle and Error Propagation
  - Case Studies and The Standish Report

- What is RE?
  - Role of Requirements

- How to do RE? -> RE Processes

Sources of Material
What is RE Really about?

Can You Stop the rain?

Rain, Rain Go Away! Go Away!
Abridada!!!

... It’s snowing!

What is it you really want?

What does the customer really want?

Lawrence Chung
Sources of Course Material
Some basic material

Introduction to RE [Davis.Ch1; LK.Ch1]

**Requirements Engineering Processes** [LK.Ch2]
- RE evolutionary process
- RE basic process
- RE in software lifecycle
- Process vs. product specifications

**Requirements Analysis, Modeling and Specification** [LK.Sec4.1 -4.2]
- Requirements Elicitation:
  - Scenario Analysis [Martin & Odell. Ch28]
- Enterprise Requirements:
  - Modeling Techniques
    - Agent-oriented enterprise modeling
    - Business modeling with UML
    - Conventional enterprise modeling techniques
  - AS-IS or TO-BE?

**Functional Requirements: Semi-formal Structural Models** [LK.Sec4.3; Davis.Ch2]
- Structured analysis
- Functional Requirements: Formal Structural Models:
  - A Formal OO-RML/Telos
    - Deficiencies of SA
    - RML/Telos Essentials
    - A Formalization
    - A Brief Survey of FMs
  - Metamodeling
    - Models, Metaclasse, Metamodels
    - Metamodels for UML and other notations

**Functional Requirements: Behavioral Models** [Davis.Ch4]
- Decision-oriented
- State-oriented
- Function-oriented behavioral models

**Non-Functional Requirements** [CNYM, 2000; LK.Ch5; Davis.Ch6]
- Why NFRs
- What – definitions and classifications
- How – product- and process-oriented approaches

Another possible topic: **Model Checking**
Parts of Lecture Notes Come From

- Plus other references as in the syllabus
- Plus some selected articles (on the next slide)
- Plus articles and web resources as indicated in individual modules
Some selected articles

Some selected articles

Some Questions

Trials and Errors: Why Science Is Failing Us
http://www.wired.com/magazine/2011/12/ff_causation/all/1

(reductionist vs. causalist?)

1 + 1 = 2?

Do stakeholders fall down from the sky when you need them?

Is my pain your pleasure?
Why RE in SysEng?

System Engineering = Engineering System

“A system (from Latin systēma, in turn from Greek σύστημα systēma, "whole compounded of several parts or members", literary "composition") is a set of interacting or interdependent components forming an integrated whole.

A system is a set of elements (often called ‘components’ instead) and relationships which are different from relationships of the set or its elements to other elements or sets.

Most systems share common characteristics, including:
• structure, defined by components/elements and their composition;
• behavior, which involves inputs, processing and outputs of material, energy, information, or data;
• interconnectivity: the various parts of a system have functional as well as structural relationships to each other.
• some functions or groups of functions.

The term system may also refer to a set of rules that governs structure and/or behavior.
Why RE in SysEng?

System Engineering = Engineering System

“Engineering is the discipline, art, skill and profession of acquiring and applying scientific, mathematical, economic, social, and practical knowledge, in order to design and build structures, machines, devices, systems, materials and processes.”
Why RE in SysEng?

System Engineering = Engineering System

Examples of a system = ????

Is the wikipedia definition of “system” good enough?
System Engineering = Engineering System

Requirements Engineering is raising and answering questions:
- Why do we need a System?
- What should a System be like?
- How do we go about building a System?

A variety of RE: RE for software system, RE for hardware, RE for enterprise, …