Software Design

Introduction
Software Design

- **How** to implement the **what**.
- *Requirements Document (RD)* is starting point.
- Software design is a highly-creative activity.
- Good designers are worth their weight in gold!
  - Highly sought after, head-hunted, well-paid.
- Experience alone is not enough:
  - creativity, “vision”, all-around brilliance required.
Some consider software design to be a “black art”:
- difficult to prescribe how to do it
- hard to measure a good design objectively
- “I know a good design when I see it.”
Requirements Engineering: An Overview

- **Basic goal**: To understand the problem as perceived by the user.
- Activities of RE are **problem oriented**.
  - Focus on *what*, not *how*
  - Don’t cloud the RD with unnecessary detail
  - Don’t pre-constrain design.
- After RE is done, do **software design**:
  - solution oriented
  - *how* to implement the *what*
Requirements Engineering: An Overview

- Key to RE is good communication between customer and developers.
- Work from Requirements Document as guide.
Requirements Engineering

• Basically, it’s the process of determining and establishing the precise expectations of the customer about the proposed software system.
The Two Kinds of Requirements

- **Functional**: The precise tasks or functions the system is to perform.
  - *e.g.*, details of a flight reservation system

- **Non-functional**: Usually, a constraint of some kind on the system or its construction
  - *e.g.*, expected performance and memory requirements, process model used, implementation language and platform, compatibility with other tools, deadlines, ...
The Purpose of RE

- Raw user requirements are often:
  - vague
  - contradictory
  - impractical or impossible to implement
  - overly concrete
  - just plain wrong

- The purpose of RE is to get a usable set of requirements from which the system may be designed and implemented, with minimal “surprises”.
The RE Process

- Requirements Analysis
  - produces
  - System Models
    - leads to
    - Requirements Definition
      - leads to
      - Requirements Specification
        - included in
        - Requirements Specification
          - leads to
          - Software Specification
            - Software Specification
              - Requirements Document
                - Software Design (Introduction)
The Requirements Document

- The official statement of what is required of the system developers.
  - Includes system models, requirements definition, and requirements specification.
  - Not a design document.
  - States functional and non-functional requirements.
- Serves as a reference document for maintenance.
Requirements Document

“Requirements”

• Should be easy to change as requirements evolve.
• Must be kept up-to-date as system changes.
The Requirements Document Should State ...

• Foreseen problems:
  – “won’t support Win-3.x apps”
• Expected evolution:
  – “will port to MacOS in next version”
• Response to unexpected events/usage:
  – “if input data in old format, will auto-convert”
Requirements Document Structure

- Introduction (describe need for system)
- Functional Requirements
- Non-Functional Requirements
- System Evolution (describe anticipated changes)
- Glossary (technical and/or new jargon)
- Appendices
- Index
A Story ...

Dear Mr. Architect,

Please design and build me a house. I am not quite sure of what I need, so you should use your discretion.

My house should have between two and forty-five bedrooms. Just make sure the plans are such that bedrooms can be easily added or deleted. When you bring the blueprints to me, I will make the final decision of what I want. Also bring me the cost breakdown for each configuration so that I can arbitrarily pick one.
Keep in mind that the house I ultimately chose must cost less than the one I am currently living in. Make sure, however, that you correct all the deficiencies that currently exist in my house (the floor of my kitchen vibrates when I walk across it, and the walls don’t have nearly enough insulation in them).

Also keep in mind as you design this house that I wish to keep yearly maintenance cost as low as possible. This should mean the incorporation of extra-cost features like aluminum or vinyl siding. If you chose not to specify aluminum, be prepared to explain in detail.
A Story … (Cont’d)

Please take care that modern design practices and the latest materials are used in construction of the house. The house should be really nice. However, be alerted that the kitchen should be designed to accommodate among other things, my 1952 Gibson refrigerator.

To assure that you are building the correct house for our family, make sure that you contact each of the children and also the in-laws. My mother-in-law will have very strong feelings about how the house ought to be designed since she visits with us at least once a year. Make sure that you weigh all these options carefully and make the right decision. I, however, retain the right to override any decision you come up with.
A Story … (Cont’d)

Please don’t bother me with small details right now. Your job is to develop the overall plans for this house. Get the big picture. It is not appropriate at this time to be choosing the color of the carpet. However, keep in mind that my wife likes green.

Also do not worry at this time about acquiring resources to build this house. Your first priority is to develop detailed plans and specifications. However, once I accept these plans, I will expect to have the house under roof within 48 hours.
While you are designing this house specifically for me, keep in mind that sooner or later I will have to sell this house. It should have appeal to potential buyers. Please make sure that before you finalize the plans, there is a consensus of the population in my area that they like the features this house has.

You are advised to run up and look at my neighbor’s house he had constructed last year. We like it a great deal. It has many features that we would like to have in our new home, particularly the 75-foot swimming pool. With careful engineering I believe that you can design this into our new house without impacting the construction cost.
Please prepare a complete set of blueprints. It is not necessary at this time to do the real design since these blueprints will be used only for construction bids. Please be advised however, that any increase of cost in the future as a result of design changes will result in you getting your hands slapped.

You must be thrilled to be working on such an interesting project such as this. To be able to use new kinds of construction and to be given such freedom in your designs is something that doesn’t happen very often. Contact me as rapidly as possible with your design ideas. I am enthusiastic about seeing what you can come up with.
P.S. My wife has just told me that she disagrees with many on the instructions I’ve given you in this letter. As architect it is your responsibility to resolve these issues. I have tried in the past and have been unable to accomplish this. If you can’t handle this, I’ll have to look for a new architect.

P.P.S. Perhaps what I need is not a house at all, but a travel trailer. Please advise me as early as possible if that is the case.
**RE Summary**

- RE focuses on determining *what* the customer wants, and not *how* it will be implemented.
- RE is hard to get correct; it requires good communication skills.
- Requirements may *change* over time.
- RE requires iteration.
**RE Summary (Cont’d)**

- The customer often doesn’t have good grasp of what he wants.
- Errors made at the requirements stage are very expensive to fix later.
  - You might well implement the stated requirements correctly, but it won’t be the system the customer really wants.
Back to Software Design ...
Software Design (Introduction)

High-Level (abstract) design

1. Software architecture
2. Subsystem decomposition
3. Subsystem dependencies
4. Subsystem interfaces
5. Module or class decomposition
6. Module or class dependencies
7. Module or class interfaces
8. Data structures

Low-Level (detailed) design

Software Design (Introduction)
Top-Down vs Bottom-Up Design

- **Top-down Design:**
  - Start with a coarsely-grained view of system, and repeatedly refine components until you have concrete sub-components.

- **Bottom-up Design:**
  - Start with existing components and “glue” them together to get what you want.
Top-Down vs Bottom-Up Design (Cont’d)

• Top-down is the “ideal” of most design methods, but it’s rarely followed absolutely:
  – some branches of development are expanded before others are even started
  – doesn’t adequately account for reuse of existing components:
    • COTS products, libraries, previous versions of the same system.
Design Quality

- Software design “quality”, as with other ideas on quality, is an elusive concept:
- It depends on priorities of your company and the customers:
  - fastest to implement
  - easiest to implement
  - easiest to maintain, “evolve”, port
  - most efficient/reliable/robust end-product.
Discussion

- What does “quality” mean to:
  - IBM?
  - Microsoft?
  - Netscape?
  - FAA?
  - IRS?
  - Intel?
  - ...

Some Desirable Design Attributes

- **Hierarchical:** A good design should be organized into a well-designed hierarchy of components.

- **Modular:** Separate distinct concerns (data and processing) into distinct containers (i.e., subsystems, modules, and/or classes). Hide implementation details and provide clean, simple interfaces for each container.
Some Desirable
Design Attributes (Cont’d)

• **Independent:** Group similar things together; limit the amount of “special knowledge” that unrelated components may share. If you change your mind about something, the impact will be localized.
Some Desirable Design Attributes (Cont’d)

• **Simple Interfaces:** Endless flexibility adds complexity. Complex interfaces mean:
  – hard to understand by users and developers (e.g., Unix *man* page syndrome)
  – many possible variations of use
  – inconvenient to change interface in order to eliminate “bad options”.

• You can get away with “flexible interfaces” in a low-level localized setting, but the larger the scale, the simpler the interface should be.