Design

Introduction

- a meaningful engineering representation of some software product that is to be built.
- can be traced to the customer's requirements.
- can be assessed for quality against predefined criteria.

2

Introduction

- In the software engineering context, design focuses on major areas of concern:
  - data,
  - architecture,
  - interfaces and
  - components.
- The emphasis in design is on quality; this phase provides us with representation of software that can be assessed for quality.
- the only phase in which the customer's requirements can be accurately translated into a finished software product or system.

3

Introduction

- Without a proper design we risk building an unstable system
  - one that will fail when small changes are made,
  - one that may be difficult to test;
  - one whose quality cannot be assessed until late in the software process, perhaps when critical deadlines are approaching and much capital has already been invested into the product.

4

Introduction

- During the design process the software specifications are transformed into design models that describe the details of
  - data structures,
  - system architecture,
  - interface, and
  - components.
- Each design product is reviewed for quality before moving to the next phase of software development.

5

Introduction

- At the end a design specification document is produced → design models that describe the data, architecture, interfaces and components.
  - At the data and architectural levels the emphasis is placed on the patterns as they relate to the application to be built.
  - At the interface level, human ergonomics often dictate the design approach employed.
  - At the component level the design is concerned with a "programming approach" which leads to effective data and procedural designs.

6

Design Specification Models

- *Data design*

  Created by transforming the analysis information model into data structures required to implement the software.

  Part of the data design may occur in conjunction with the design of software architecture.

  More detailed data design occurs as each software component is designed.

7

Design Specification Models

- *Architectural design*

  Defines the relationships among the major structural elements of the software.

  The "patterns" then can be used to achieve the requirements that have been defined for the system.

  The constraints that affect the way in which the architectural patterns can be applied.

  It is derived from the system specification, the analysis model, and the subsystem interactions defined in the analysis model.

8

Design Specification Models

- *Interface design*

  Describes how the software elements communicate with each other, with other systems, and with human users.
- *Component-level design*

  Created by transforming the structural elements defined by the software architecture into procedural descriptions of components using information obtained from the process specification, control specification, and state transitions.

9

Design Guidelines

- The criteria for a good design:
  - exhibit good architectural structure
  - be modular
  - contain distinct representations of data, architecture, interfaces, and components (modules)
  - lead to data structures that are appropriate for the objects to be implemented and be drawn from recognizable design patterns

10

Design Guidelines

- lead to components that exhibit independent functional characteristics
- lead to interfaces that reduce the complexity of connections between modules and with the external environment
- be derived using a reputable method that is driven by information obtained during software requirements analysis

11

Design Principles

- The set of principles established to aid the software engineer in navigating the design process are:
  - *The design process should not suffer from tunnel vision*

    A good designer should consider alternative approaches. Judging each based on the requirements of the problem, the resources available to do the job and any other constraints.
  - *The design should be traceable to the analysis model*

    Because a single element of the design model often traces to multiple requirements, it is necessary to have a means of tracking how the requirements have been satisfied by the model

12

Design Principles

- *The design should not reinvent the wheel*

  Systems are constructed using a set of design patterns, many of which may have likely been encountered before.

  These patterns should always be chosen as an alternative to reinvention.

  Design time should be invested in representing truly new ideas and integrating those patterns that already exist.

13

Design Principles

- *The design should minimise intellectual distance between the software and the problem as it exists in the real world*

  That is, the structure of the software design should (whenever possible) mimic the structure of the problem domain.

- *The design should exhibit uniformity and integration*

  A design is uniform if it appears that one person developed the whole thing.

  Rules of style and format should be defined for a design team before design work begins.

14

Design Principles

- *The design should be structured to degrade gently, even with bad data, events, or operating conditions are encountered*

  Well-designed software should never "bomb".

  It should be designed to accommodate unusual circumstances, and if it must terminate processing, do so in a graceful manner.

15

Design Principles

- *The design should be reviewed to minimize conceptual (semantic) errors*

  There is sometimes the tendency to focus on minute details when the design is reviewed, missing the forest for the trees.

  The designer team should ensure that major conceptual elements of the design have been addressed before worrying about the syntax if the design model.

16

Design Principles

- *Design is not coding, coding is not design*

  Even when detailed designs are created for program components, the level of abstraction of the design model is higher than source code.

  The only design decisions made of the coding level address the small implementation details that enable the procedural design to be coded.
- *The design should be structured to accommodate change*
- *The design should be assessed for quality as it is being created*

17

Design Principles

- When these design principles are properly applied, the design exhibits both external and internal quality factors.
- External quality factors
  - factors that can readily be observed by the user
  - e.g. speed, reliability, correctness, usability
- Internal quality factors
  - relate to the technical quality, the quality of the design itself
  - important to the software engineer

18

Fundamental Design Concepts

- Each concept helps the soft ware engineer to answer the following questions:
  - What criteria can be used to partition software into individual components?
  - How is function or data structure detail separated from a conceptual representation of software?
  - Are there uniform criteria that define the technical quality of a software design?

19

Fundamental Design Concepts

- The fundamental design concepts are:
  - *Abstraction*

    allows designers to focus on solving a problem without being concerned about irrelevant lower level details (procedural abstraction - named sequence of events, data abstraction - named collection of data objects)
  - *Refinement*

    process of elaboration where the designer provides successively more detail for each design component

20

Fundamental Design Concepts

- *Modularity*

  the degree to which software can be understood by examining its components independently of one another
- *Software architecture*

  overall structure of the software components and the ways in which that structure provides conceptual integrity for a system

21

Fundamental Design Concepts

- *Control hierarchy or program structure*

  represents the module organization and implies a control hierarchy, but does not represent the procedural aspects of the software (e.g. event sequences)
- *Structural partitioning*

  horizontal partitioning defines three partitions (input, data transformations, and output);

  vertical partitioning (factoring) distributes control in a top-down manner (control decisions in top level modules and processing work in the lower level modules).

22

Fundamental Design Concepts

- *Data structure*
  representation of the logical relationship among individual data elements (requires at least as much attention as algorithm design)
- *Software procedure*
  precise specification of processing (eventsequences, decision points, repetitive operations, data organization/structure)
- *Information hiding*
  information (data and procedure) contained within a module is inaccessible to modules that have no need for such information

23

Summary

- Software design can viewed as a process and a model.
- In general, the process be broken down into three broad stages:
  - *Architectural design*
    specifications are analysed and the desired module structure is produced
  - *Detailed design*
    each module is designed in detailed and specific algorithms and data structures are selected.
  - *Design testing*
    this is an activity that must be continuously conducted in parallel with all software production activities.

24

Now, in your project ...

- Design document, containing
  - Introduction
    - Problem Definition, Project Scope and Goals, Design Constraints,...
  - Overall Architecture
    - An informal drawing which will be explained in detail.
    - Also subsystems must be mentioned.
  - Data Design
    - ER Diagram, Data Dictionary (not much detailed)

25

Now, in your project ...

- Design Diagrams
  - Class Diagram
    - Should be well-thought and detailed.
    - Explanation of the diagram will be useful.
  - Activity-Sequence-Collaboration
    - Should only be drawn for 2-5 important functionalities
    - Good Example: Creating a new question for test management system
    - Bad Example: User Login Screen

26

Now, in your project ...

- User Interface Design
  - Some User Interfaces coded without functionality
    - for only some important functionalities
- Schedule
  - Updated version

27