SWE 530: Software Design Process

Describing a Design Solution

Dr. H. Birkan YILMAZ
Department of Computer Engineering
Boğaziçi University

(birkan.yilmaz@bogazici.edu.tr)

Adapted from slides of Dr. Albert Ali Salah & Başak Aydemir

Recap

- Software quality concepts are for assessing the static structure and the dynamic behaviour of the system.
- The ultimate goal of quality must be that of **fitness for purpose**; the criteria will be both problem-dependent and domain-dependent.
- Abstraction is an important tool, but it makes it difficult to make any direct product measurements during design.
- Technical design reviews can provide a valuable means of obtaining and using domain knowledge (for both product **and** process)

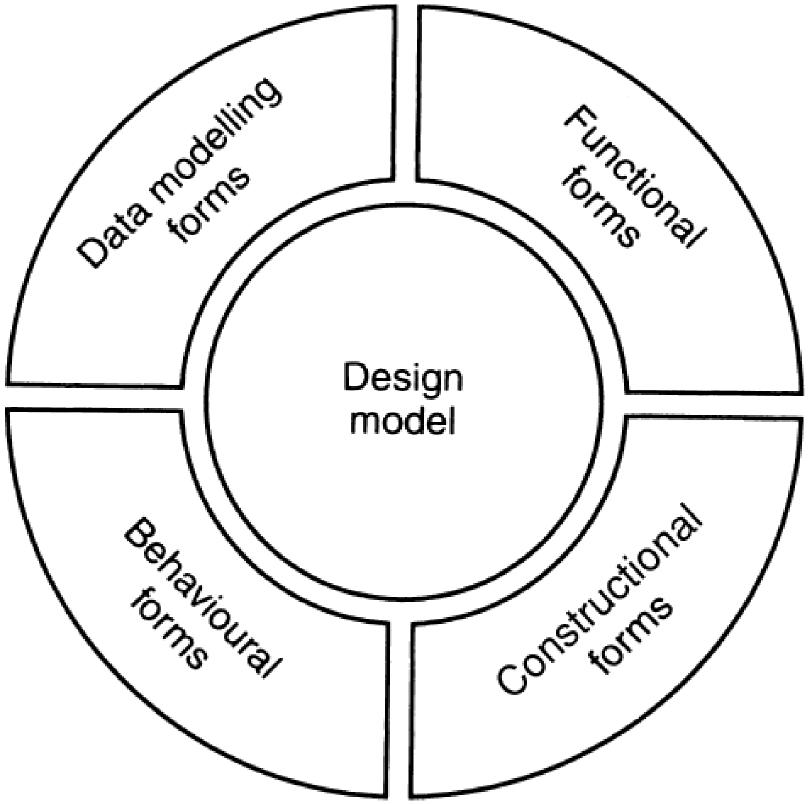
Summary

- Representing abstract ideas
- Design viewpoints for software
- Forms of notation

Representation

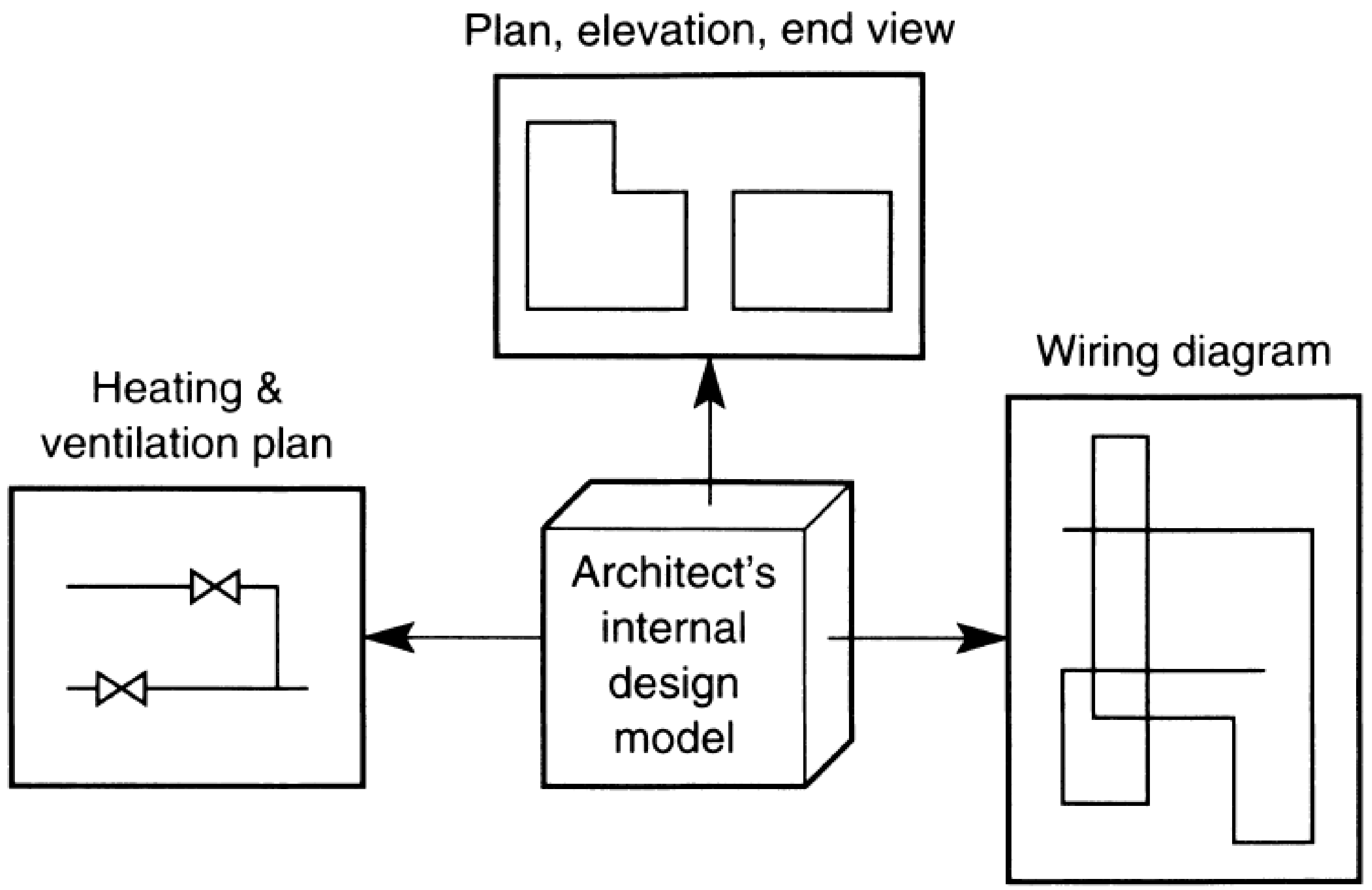
- Representation is a particular abstraction
- Can be associated with problem models and solution (program) forms
- Linked to the concept of “viewpoint”
- Representation is not the design itself!

Four principal viewpoints



The four principal viewpoints as projections from the design model.

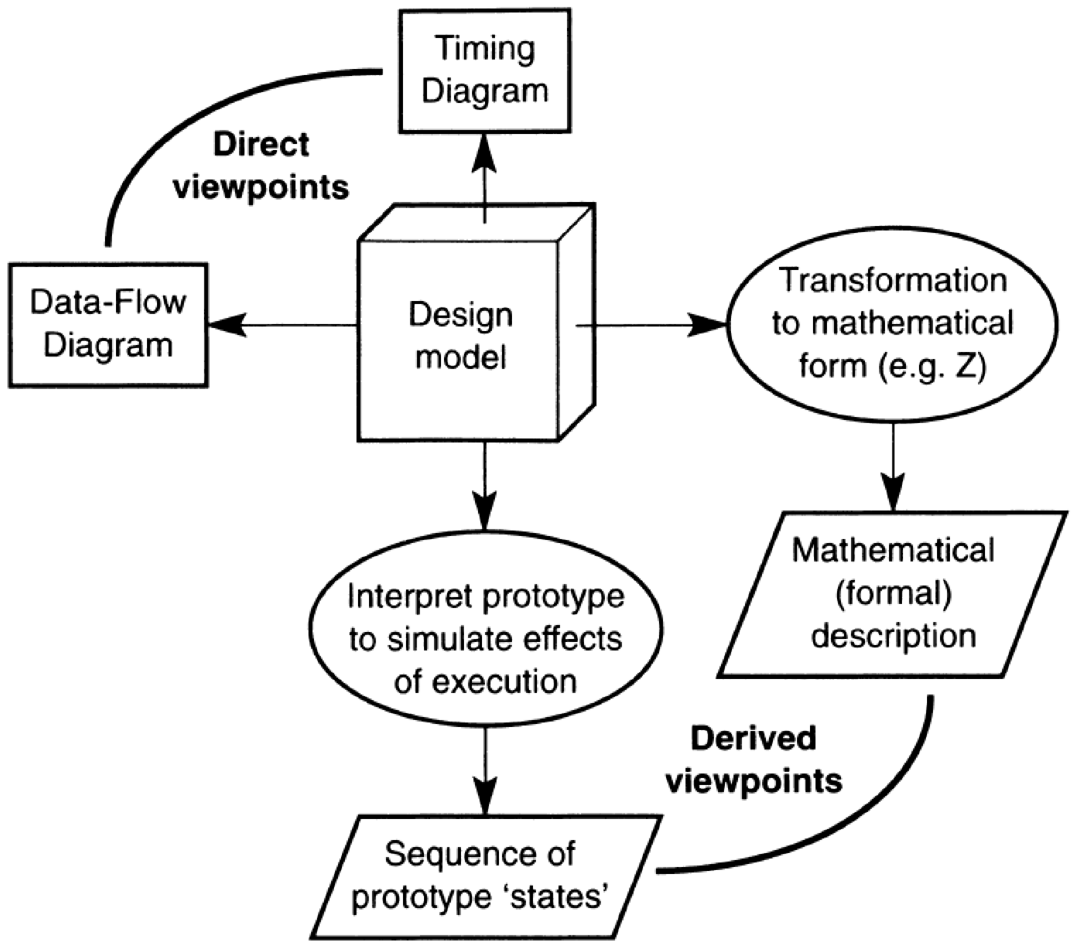
Representations


Plan, elevation, end view
Heating & ventilation plan
Architect's internal design model
Wiring diagram

Design viewpoints

- System-oriented properties:
  - Describe the dynamic behavior of the system
  - Emphasize flow of data around the system
  - Sequencing of operations…
- Solution-oriented properties:
  - “Constructional” issues like packaging, procedure hierarchy, and data organization
  - Focus on static design attributes

Derived viewpoints


Timing Diagram
Direct viewpoints
Data-Flow Diagram
Design model
Transformation to mathematical form (e.g. Z)
Mathematical (formal) description
Interpret prototype to simulate effects of execution
Derived viewpoints
Sequence of prototype 'states'

Constructional forms

- Concerned with describing how software structuring forms are used in the final system.
- Describes the outcome of the design process.
- Includes:
  - Data specification
  - Threads of execution
  - Packaging constructs
- Models static structure as opposed to dynamic, run-time behavior

Constructional forms

- Specifies relationships and dependencies between the elements of the system.
- Examples are:
  - Data flow
  - Invocation
  - Uses hierarchy
- Depends on the “architectural” form of the eventual system.

Behavioral forms

- Concerned with causal issues, connecting “events” to “responses”
- More abstract than constructional forms
- One (abstract) behavior can be spread across a number of physical elements
- Typical example is the finite state machine (and its derivatives)

Behavioral forms

- Representing temporal aspects:
  - Sequencing
  - Fixed-interval
  - Constraints (more difficult to capture!)
- Can use black-box (whole system) or white-box (chains of events and actions) modelling

Functional forms

- Problem-driven definitions of what the system does.
- Better for the algorithmic aspects.
- For example: Descriptions of run-time behavior of program elements, such as subprograms.

Data-modelling forms

- Models data-related issues, like:
  - type (classes, inheritance, etc.)
  - sequence (trees, lists, etc.)
  - form
- More related to analysis than design, and with white-box approaches
- Detailed representations

Forms of notation

- Three basic components for constructing representations:
  - Text
  - Diagrams
  - Mathematical expressions

Notations 1: Text

- Include structured forms (lists, tables)
- Summary (itemized?)
- Indentation, bold, italic, underline can be used for emphasis and organization
- Cons:
  - Easy to obscure information with text
  - Prone to ambiguity

Notations 2: Diagrams

- Good for concepts about hierarchy, position, flow of information and other forms of relationship between abstract objects
- Should be drawable with pencil, or on board
- Can be hierarchically organized
- Flowcharts:
  - Places too much emphasis on sequence
  - Defines solution via operations, rather than problem and related structures (not recommended)

Notations 3: Mathematical notations

- Abstraction + Lack of ambiguity
- Advanced level; requires additional training
- Complexity of notation

Summary

- the roles of representations in capturing, explaining, and checking design information;
- the concept of a viewpoint of a design model, as a means of capturing a particular set of
- design properties, and as projected through the use of a representation;
- the principal classes of direct design viewpoints – the constructional, behavioral, functional, data-modelling forms;
- the use of text, diagrams, and mathematical expressions as the three basic forms in constructing design representations.

Questions?

Exercise 1

- Draw a flowchart of “making Turkish Coffee”
- What design attributes does flowchart capture, and hence what viewpoint on the design model does it provide?

Exercise 2

- Suggest how you might represent the following viewpoints using in turn: text on its own; a diagram; a mathematical form:
  1. the hierarchy of pages in a website;
  2. the program units (procedures) that make use of a particular data type in a program.