SWE 530: Software Design Process

Some Design Representations

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

- We examined some of the principal ways in which software design knowledge and expertise can be codified and transferred:
  - the reasons for transferring design knowledge,
  - the role of the concept of architectural style in providing a framework and a vocabulary for top-level design ideas,
  - the use of design methods to codify design practices and strategies;
  - the rationale for using design patterns and their ability to describe the core features of reusable design solutions.

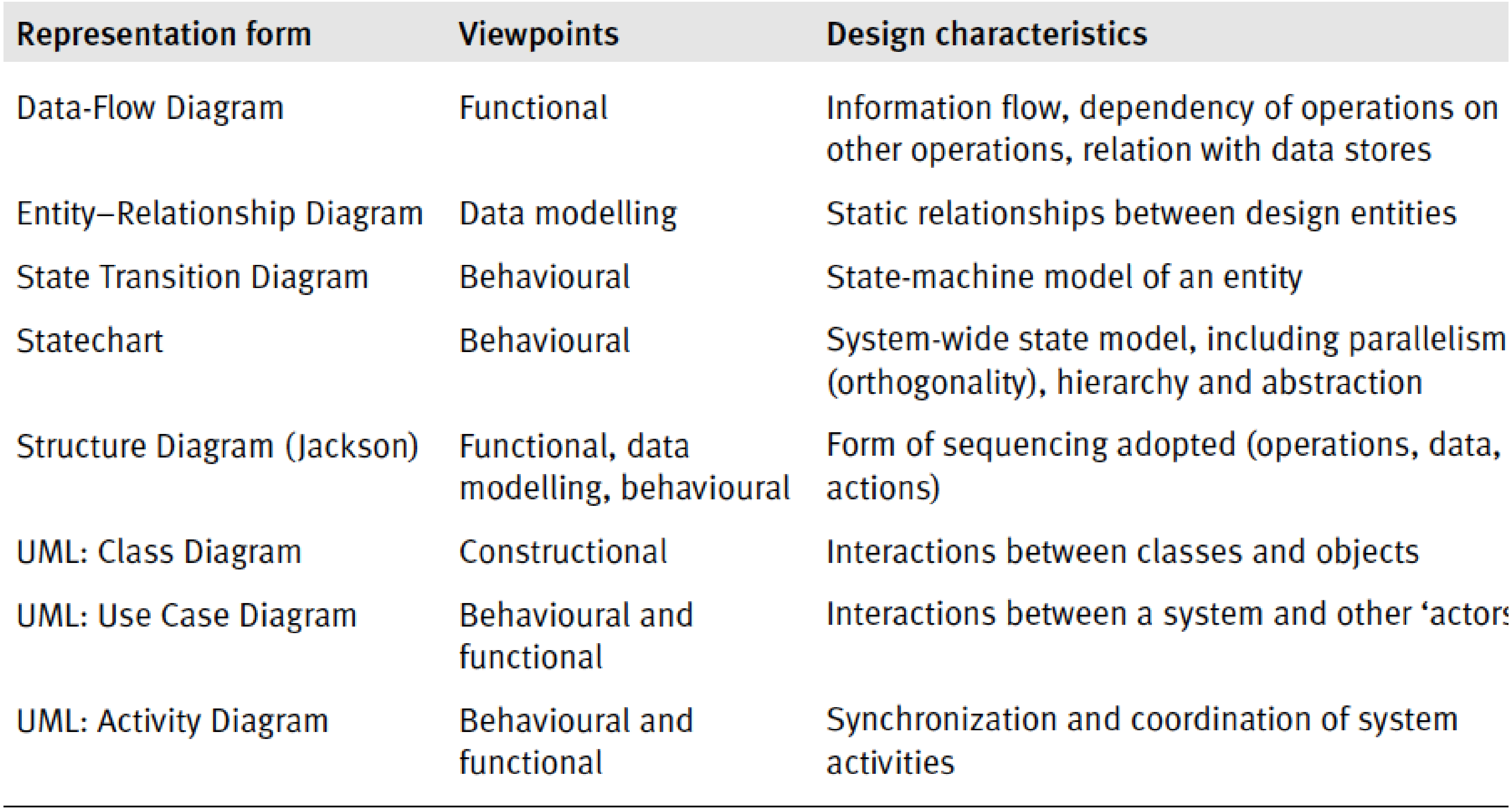
Summary

- The problem of selecting forms
- Black box notations
- White box notations
- Developing a diagram

Range of forms

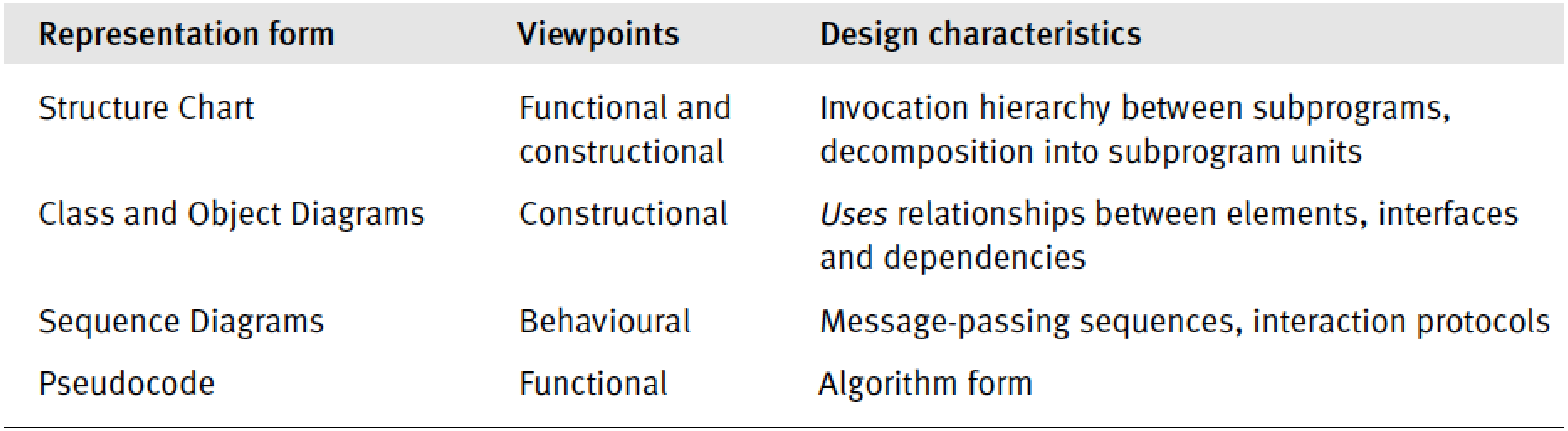
- The roles and forms of representation range across:
  - form: including textual and diagrammatical forms of notation
  - viewpoint: constructional, behavioural, functional and data-modelling viewpoints
  - use: in terms of the form’s role during the phases of design, the type of problem domain in which it might be appropriate, and the extent to which it is used

Selection of black box forms



Representation form	Viewpoints	Design characteristics
Data-Flow Diagram	Functional	Information flow, dependency of operations on other operations, relation with data stores
Entity–Relationship Diagram	Data modelling	Static relationships between design entities
State Transition Diagram	Behavioural	State-machine model of an entity
Statechart	Behavioural	System-wide state model, including parallelism (orthogonality), hierarchy and abstraction
Structure Diagram (Jackson)	Functional, data modelling, behavioural	Form of sequencing adopted (operations, data, actions)
UML: Class Diagram	Constructional	Interactions between classes and objects
UML: Use Case Diagram	Behavioural and functional	Interactions between a system and other ‘actors
UML: Activity Diagram	Behavioural and functional	Synchronization and coordination of system activities

Selection of white box forms



Representation form	Viewpoints	Design characteristics
Structure Chart	Functional and constructional	Invocation hierarchy between subprograms, decomposition into subprogram units
Class and Object Diagrams	Constructional	*Uses* relationships between elements, interfaces and dependencies
Sequence Diagrams	Behavioural	Message-passing sequences, interaction protocols
Pseudocode	Functional	Algorithm form

Black box forms

- A black box notation is one that is concerned with the **external properties** of the elements of a design model. That is, it is used to describe **what** an element will do, rather than how to do it.
- Data-Flow Diagram (DFD), Entity-Relationship Diagram (ERD), State Transition Diagram (STD), Statechart, Jackson Structure Diagram, UML forms.

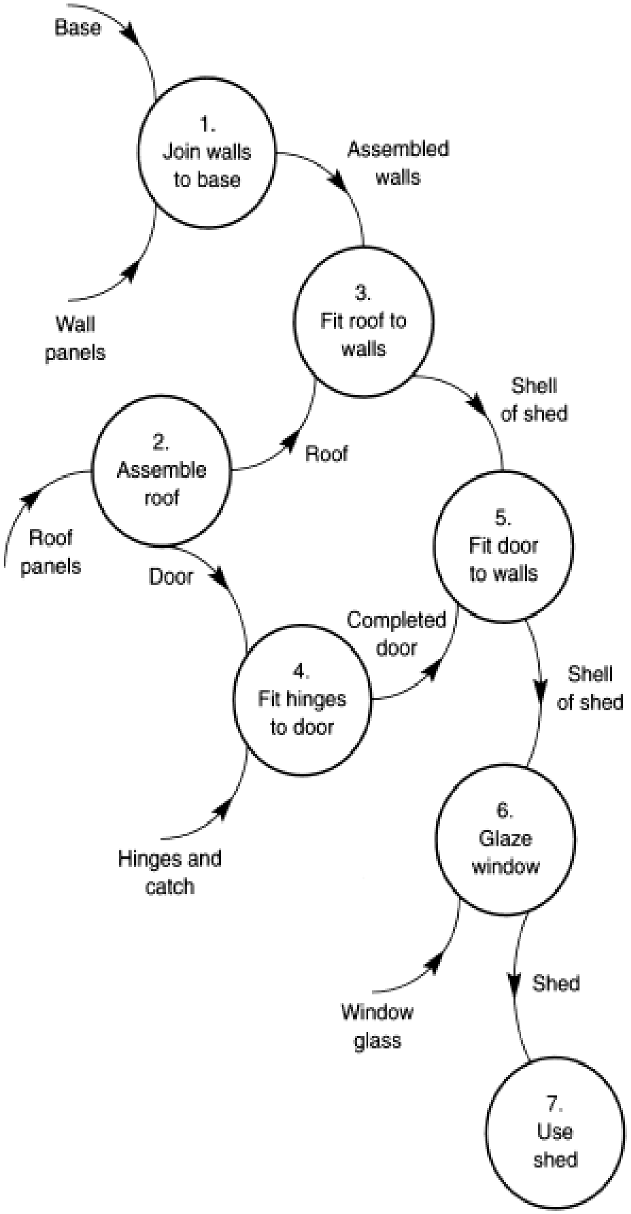
Data-Flow Diagram

- Problem oriented view of the system
- Each element modifies the information flowing into the element
- Pre-dates the computer era!
- Very effective to describe a **process**, stresses dependencies and prerequisites

Data-Flow Diagram COMPONENTS

- It has four elements:
  - the **circle** (or, as it is popularly termed, the bubble), which is used to denote an operation, and is labelled with a brief description of the operation;
  - the **box**, used to denote an external source or sink of information;
  - the **parallel bars**, to denote a data store or file;
  - the **arc**, used to denote the flow of information between the other three components.

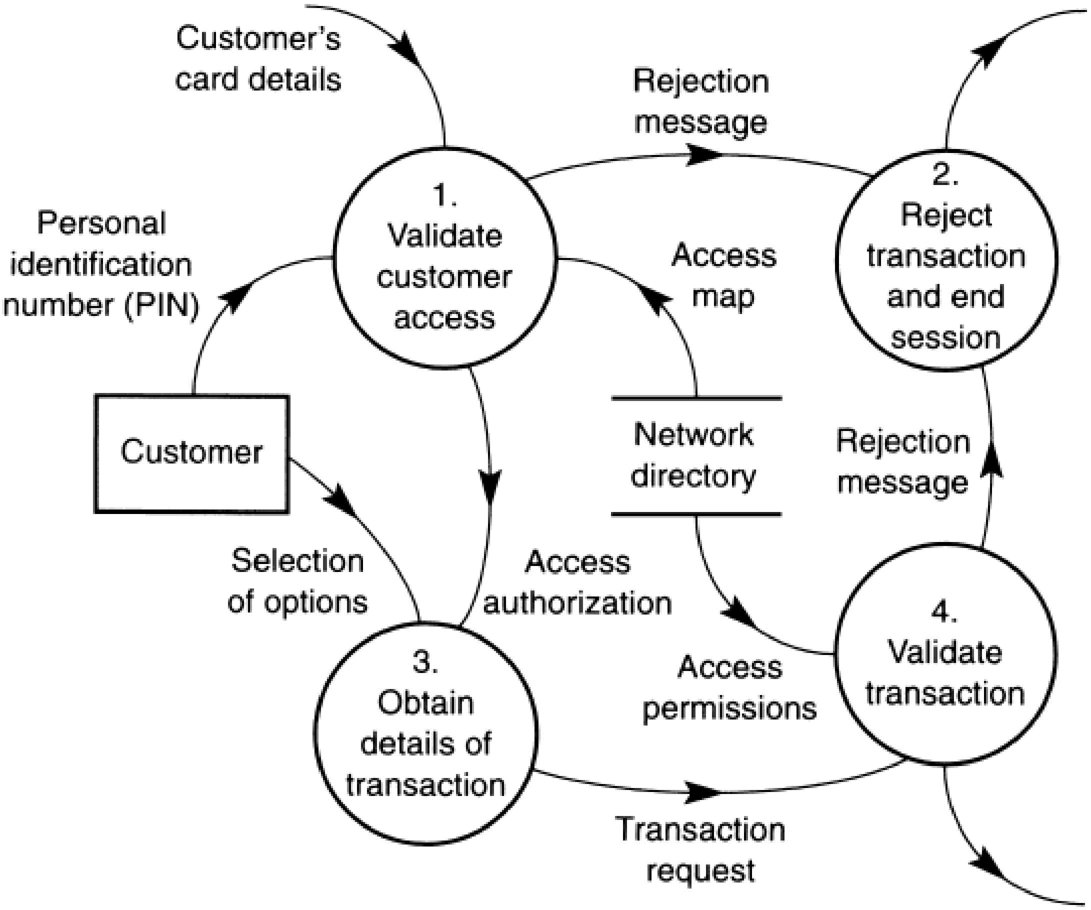
Data-Flow Diagram

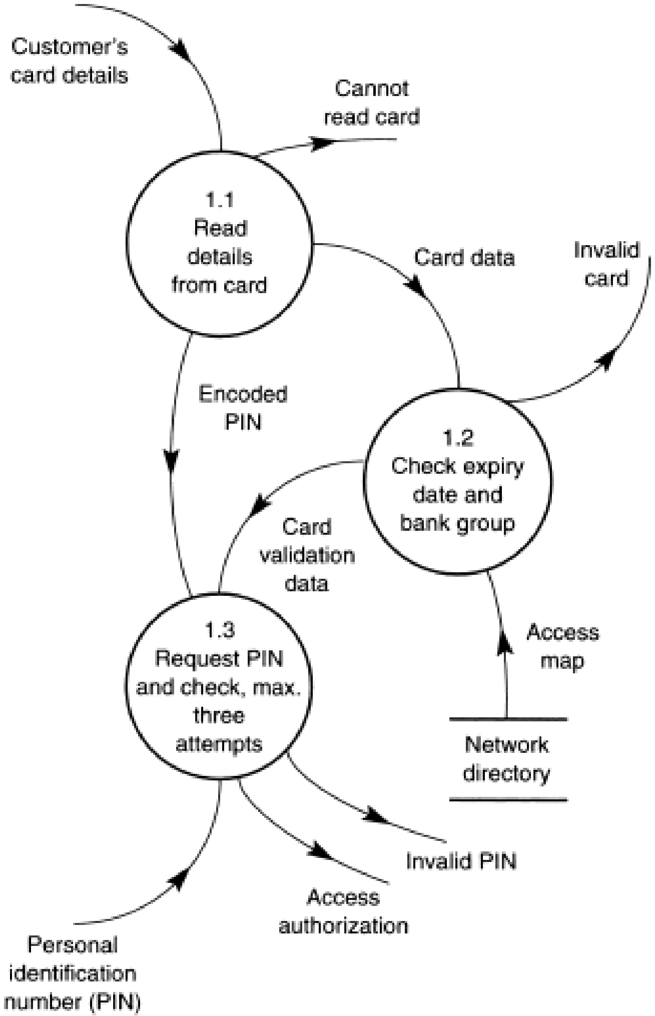


Activity

- Design a top-level DFD for an ATM machine
- Use a single “transaction” as an abstract operation
- What are the:
  - Operations (bubbles)
  - Sources and sinks of information (boxes)
  - Data stores (parallel bars)
  - Flow of information (arcs)?

DFD examples: ATM





Data-Flow Diagram

- Can be expanded in a hierarchical fashion
- Is not concerned with the control logic
- Changes at one level can cause inconsistencies with other levels…
- Special drawing tools enable automatic consistency checking.

The DFD viewpoint

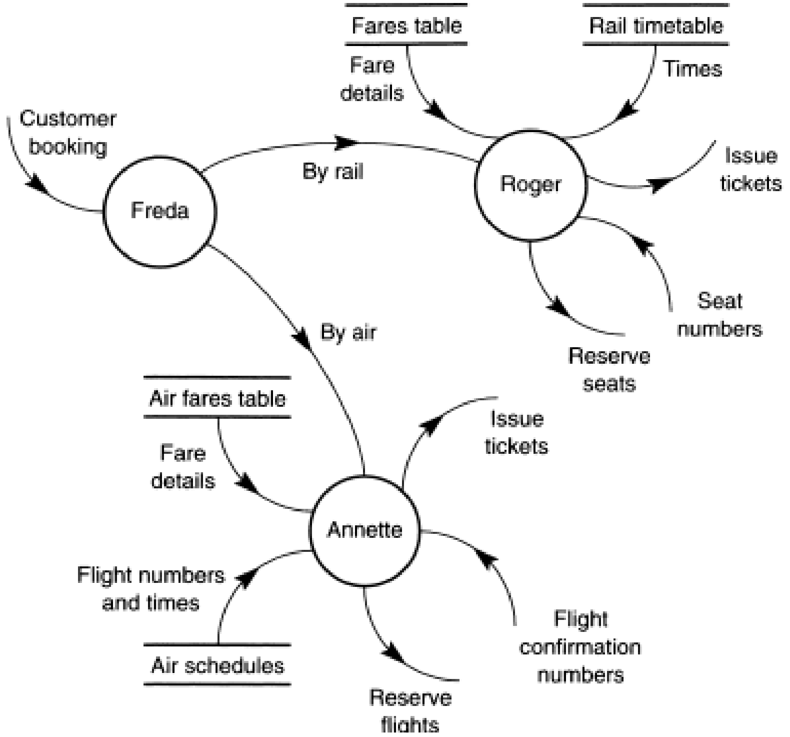
- Describes the architecture in terms of functions
- Does not specify whether the operations are serially performed, or in parallel.
- Some design methods use DFD to describe parallel operations, but the convention is to assume sequential order.

Using the DFD

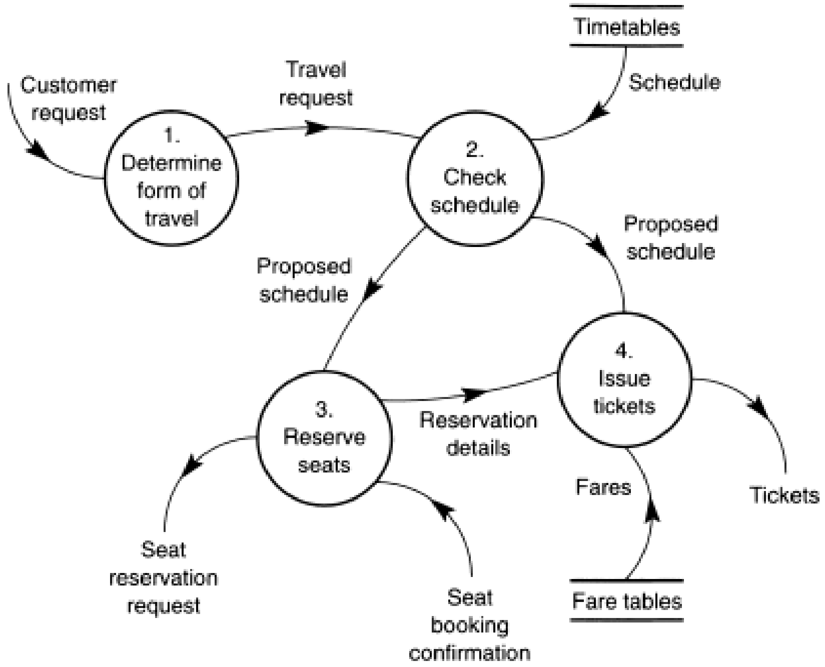
- Widely used for initial modeling (analysis)
- It models the problem, and the user can easily understand its logic
- De Marco’s distinction:

1. logical DFD: what is being done to data
2. physical DFD: who is doing it (physical entities)

Logical vs. Physical DFD

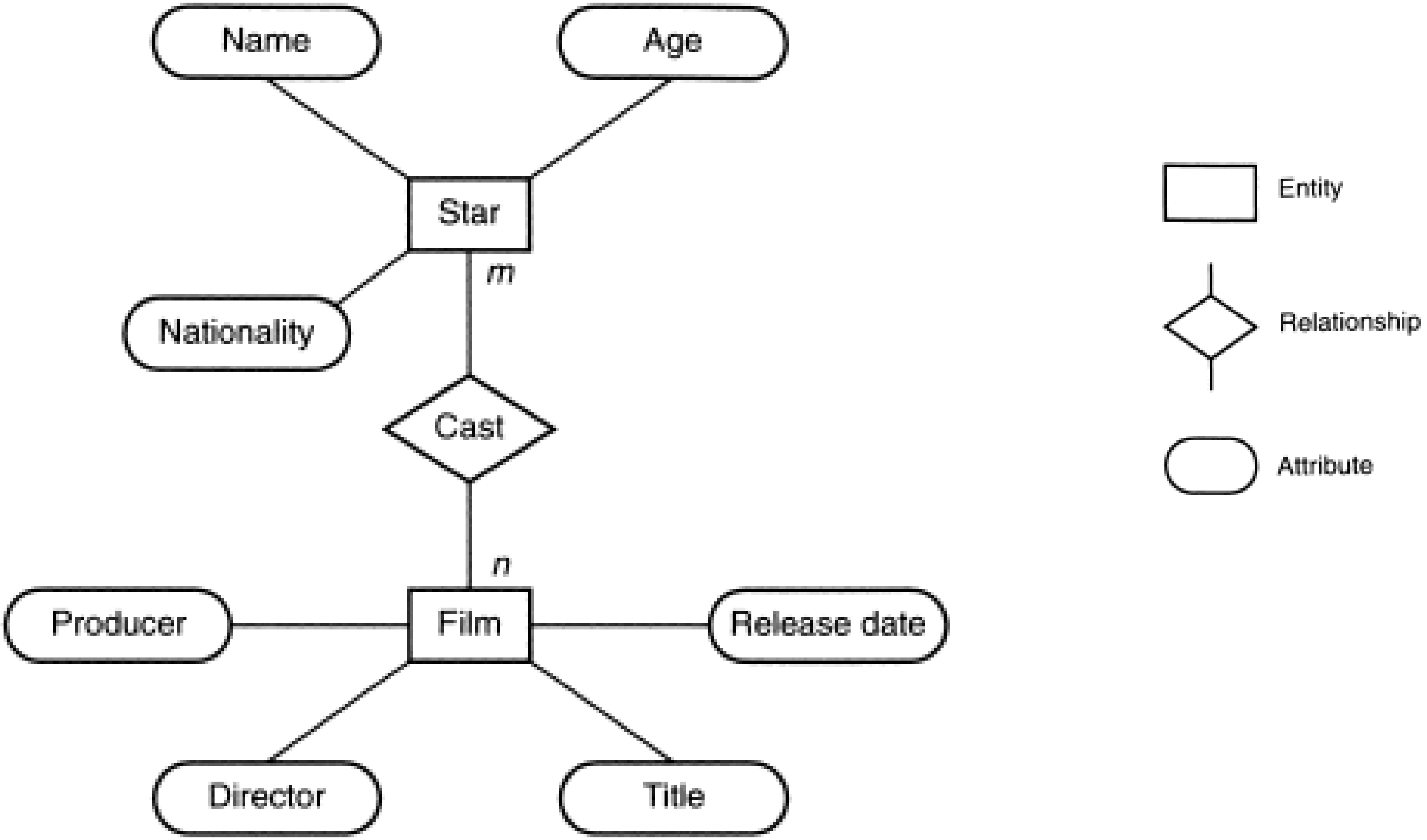


physical DFD



logical DFD

Entity-Relationship Diagram



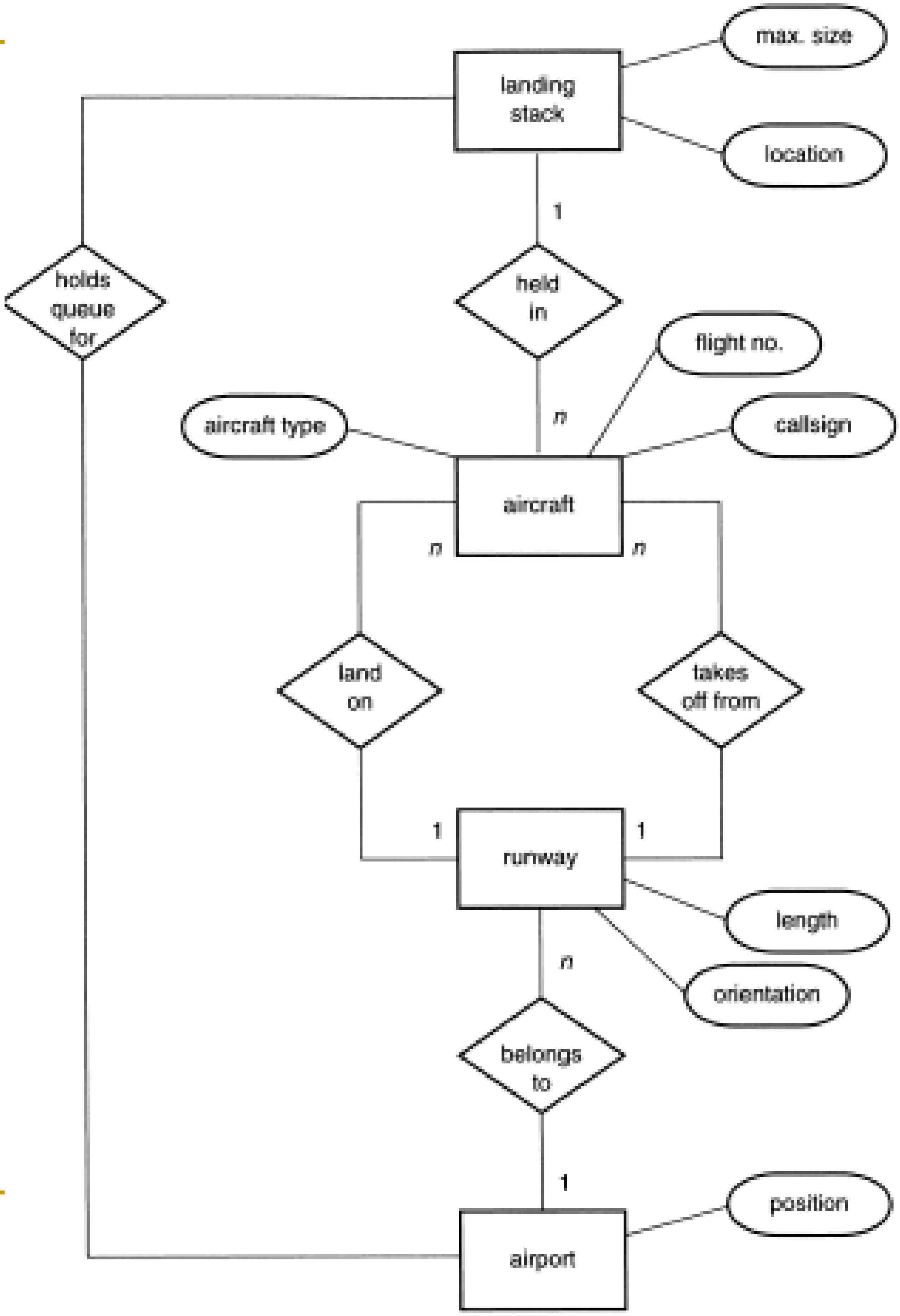
Entity-Relationship Diagram

- Entities may be connected via multiple relationships
- Attributes may be composite
- Relationships may be one to one (1 to 1), one to many (1 to n), or many to many (m to n)
  - Books held in a library?
  - Authors having written books?

Activity

- Design the ERD of an air traffic control system.
  - The entities are: airport, runway, aircraft, landing stack
  - Relations are: “belongs to”, “lands on”, “takes off from”, “held in”, and “holds queue for”
  - Each entity has 1 or 2 attributes.

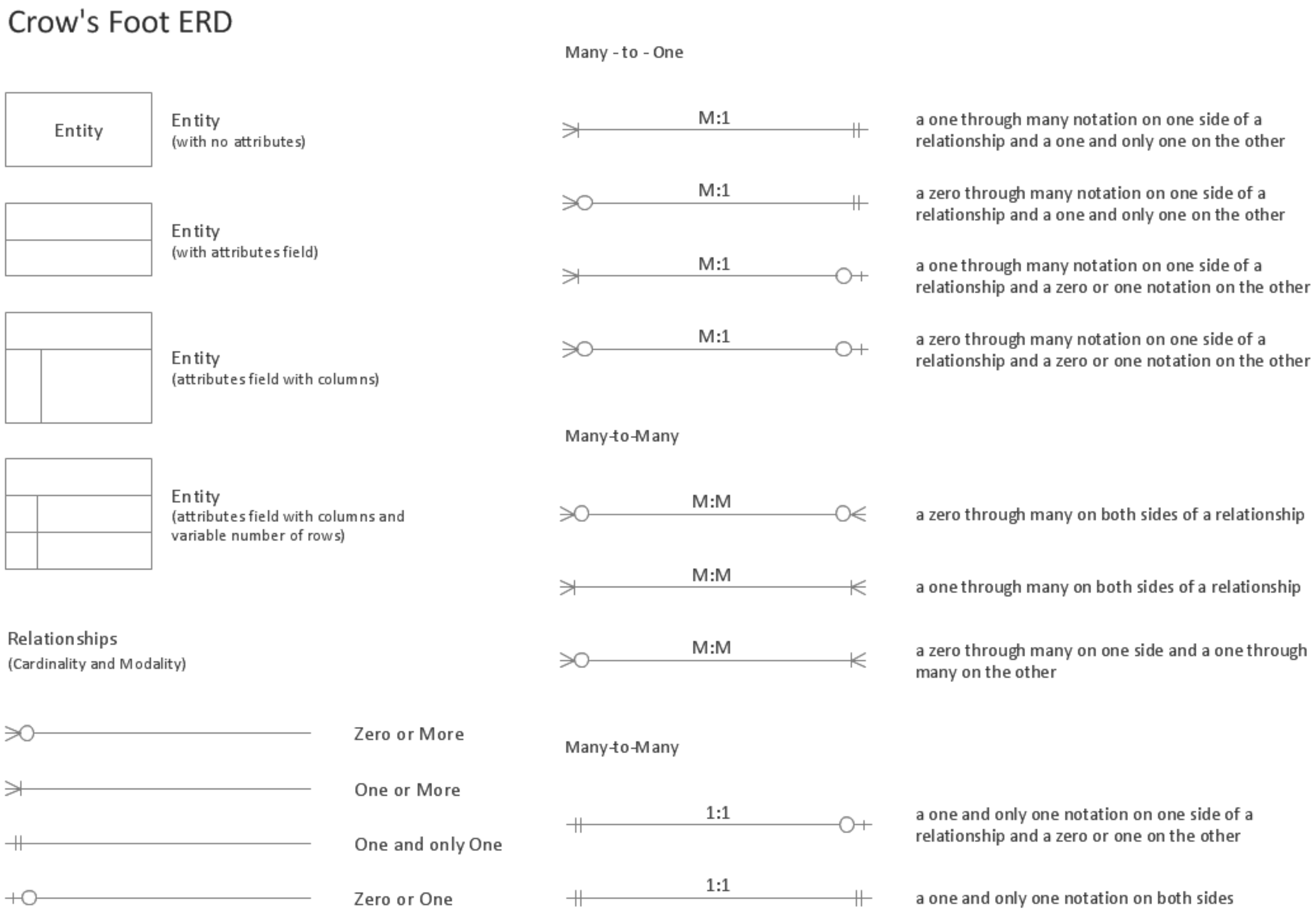
Entity-Relationship Diagram



ERD Viewpoint and Use

- ERD Viewpoint:
  - Provides a data-modelling viewpoint of a system
- Use of the ERD:
  - Developing relational **database** modelling schemas
  - Plays a role in SSA/SD (Structured Systems Analysis and Structured Design)

Different ERD Forms



Crow's Foot ERD

Entity
(with no attributes)

Entity
(with attributes field)

Entity
(attributes field with columns)

Entity
(attributes field with columns and variable number of rows)

Relationships
(Cardinality and Modality)

Zero or More

One or More

One and only One

Zero or One

Many - to - One

a one through many notation on one side of a relationship and a one and only one on the other

a zero through many notation on one side of a relationship and a one and only one on the other

a one through many notation on one side of a relationship and a zero or one notation on the other

a zero through many notation on one side of a relationship and a zero or one notation on the other

Many-to-Many

a zero through many on both sides of a relationship

a one through many on both sides of a relationship

a zero through many on one side and a one through many on the other

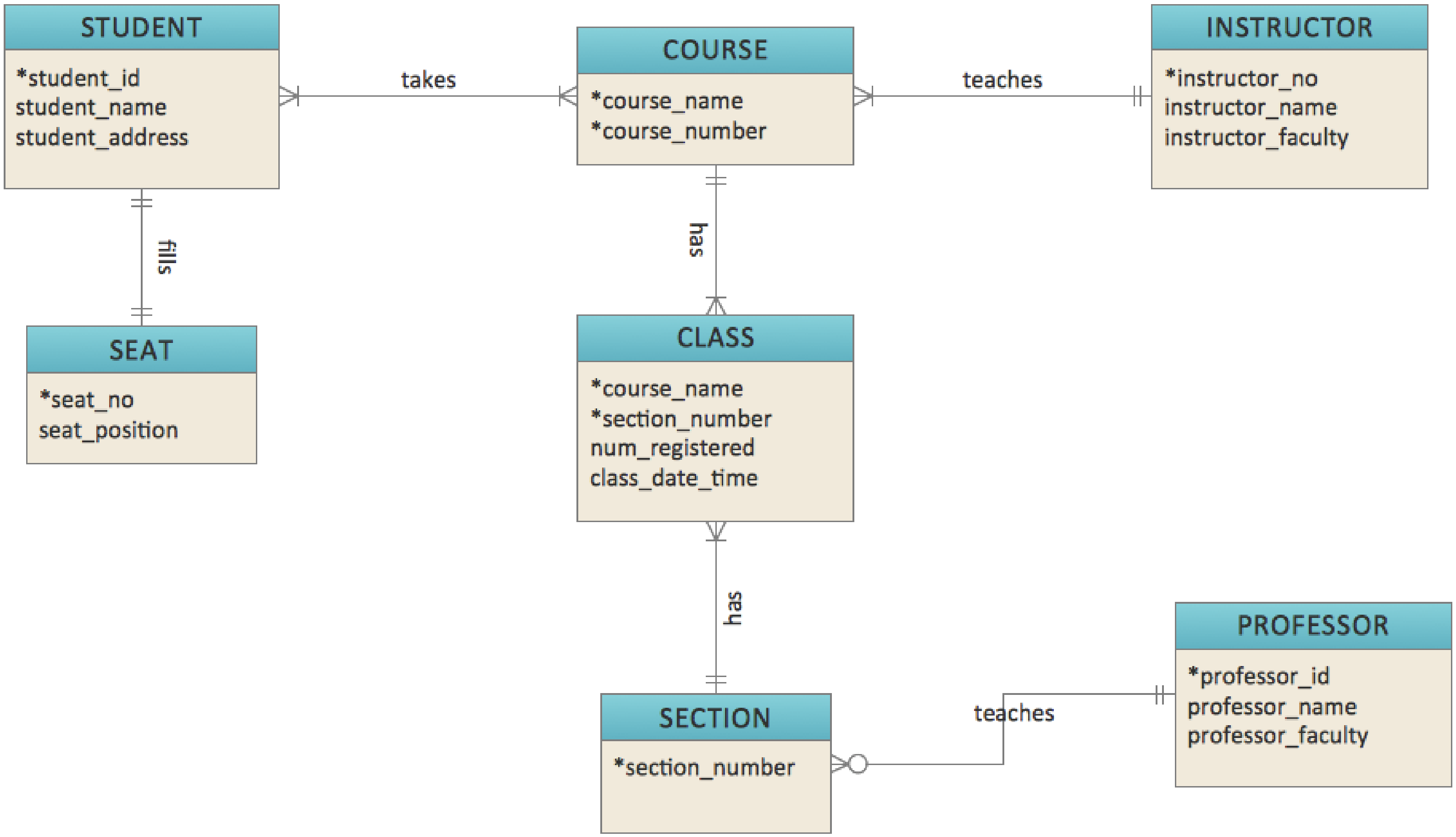
Many-to-Many

a one and only one notation on one side of a relationship and a zero or one on the other

a one and only one notation on both sides

http://www.conceptdraw.com/solution-park/diagramming-ERD

Crow's Foot Notation (more useful)

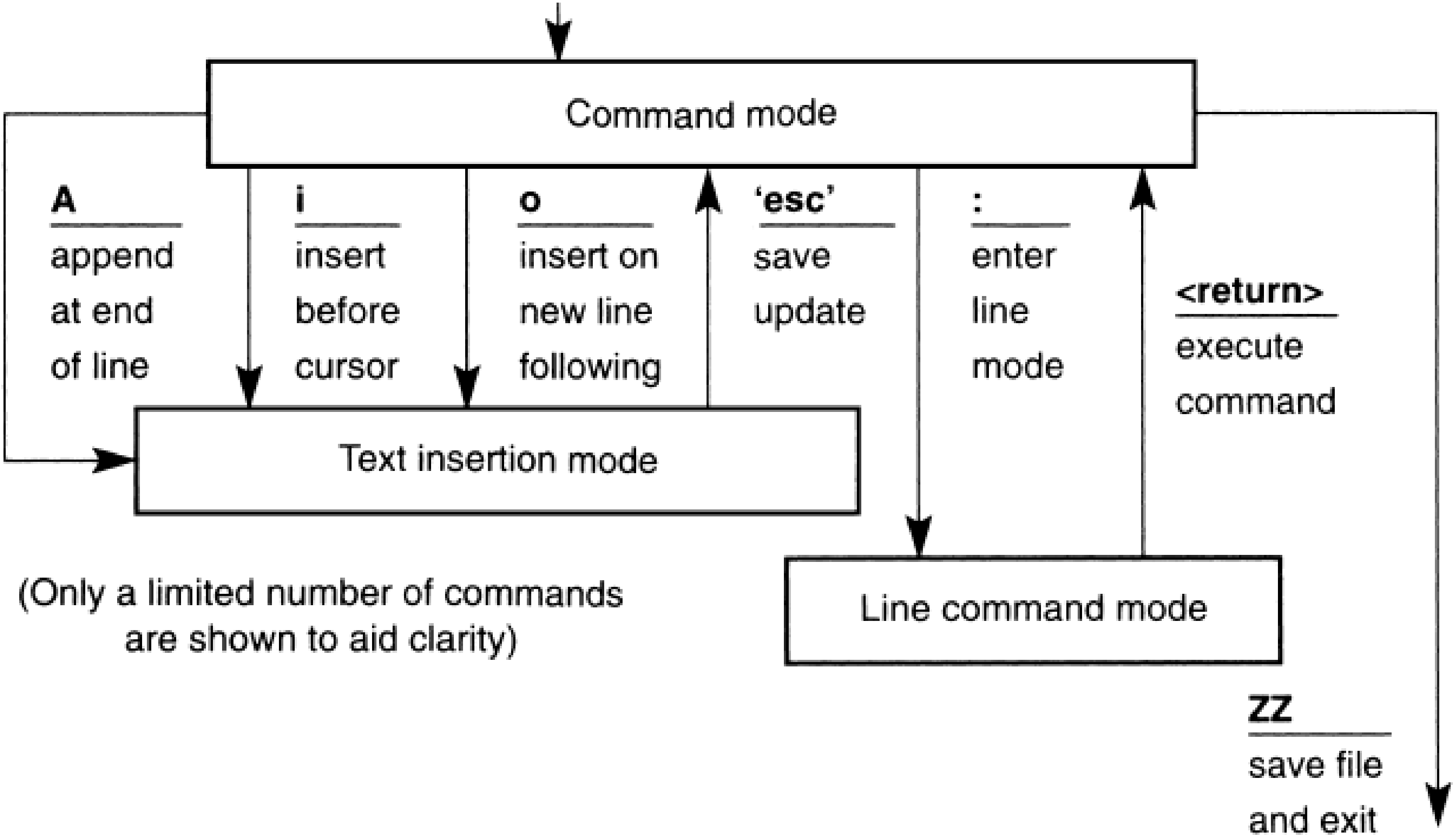


http://www.conceptdraw.com/solution-park/diagramming-ERD

State Transition Diagram (STD)

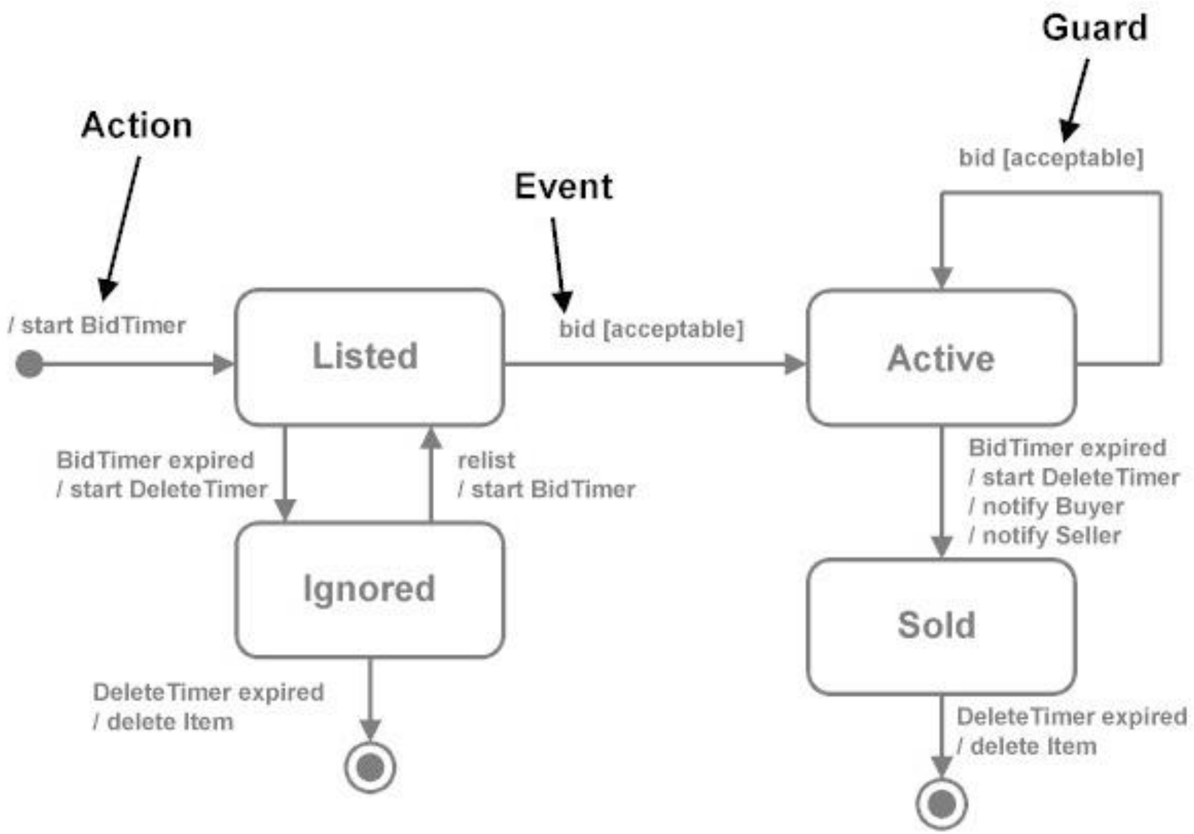
- The form of STD used in SSA/SD has:
  - The **state** represents an externally observable mode of behaviour, and is represented by a **box**, with a text label
  - The **transition** is described by an **arrow**, and identifies a ‘legal’ change of state
  - The transition **condition**
  - The transition **action** describes the actions that arise as a result of the transition

State Transition Diagram



(Only a limited number of commands are shown to aid clarity)

State Transition Diagram



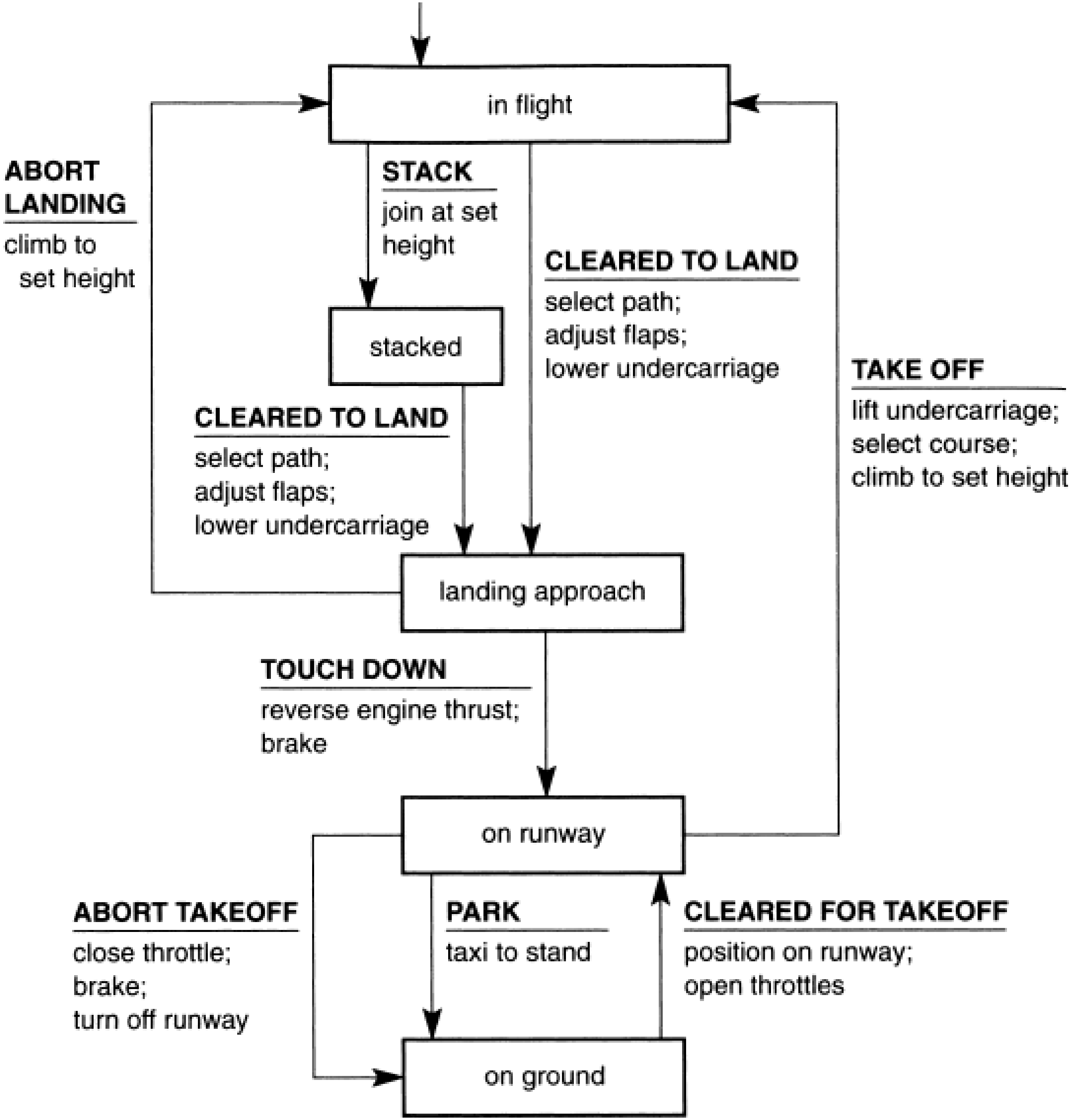
Viewpoint and Use of STD

- Captures the **behaviour** of the system
- The transitions are not sequenced; they exist as conditionals
- Major role of STD is in modelling problem entities, and in real-time needs of the system
- There can be many states and many transitions
- Not easy to implement a hierarchy

Activity

- Design the STD for describing the behaviour of an **aircraft** in an air traffic control zone
- Define
  - States
  - Events
  - Actions

State Transition Diagram



in flight

ABORT LANDING
climb to set height

STACK
join at set height

stacked

CLEARED TO LAND
select path;
adjust flaps;
lower undercarriage

CLEARED TO LAND
select path;
adjust flaps;
lower undercarriage

TAKE OFF
lift undercarriage;
select course;
climb to set height

landing approach

TOUCH DOWN
reverse engine thrust;
brake

on runway

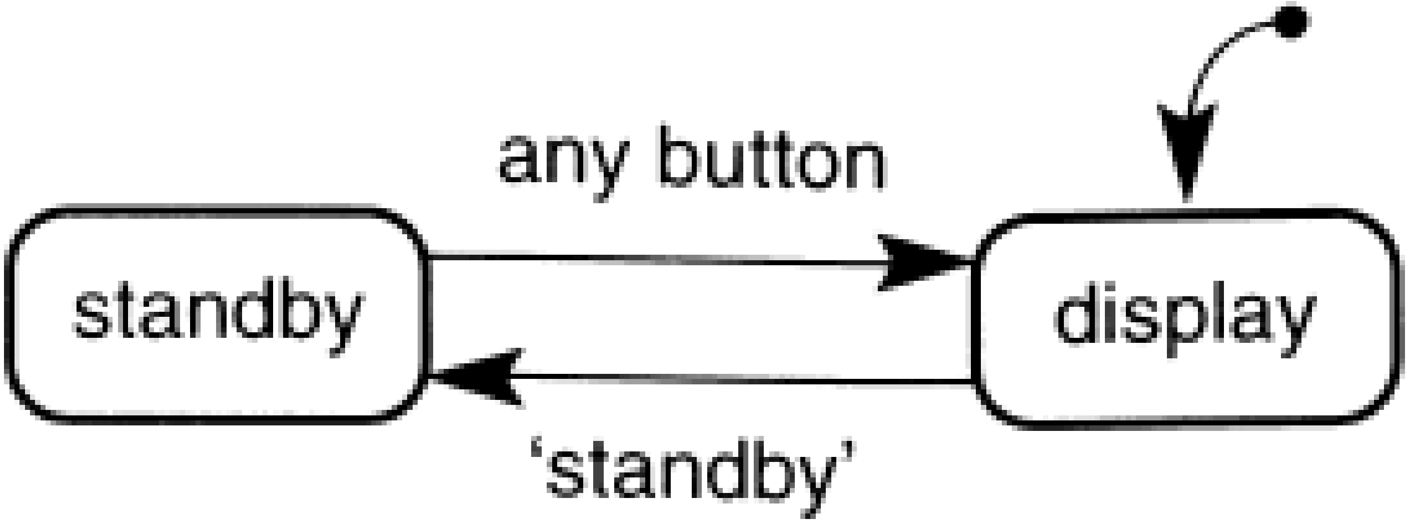
ABORT TAKEOFF
close throttle;
brake;
turn off runway

PARK
taxi to stand

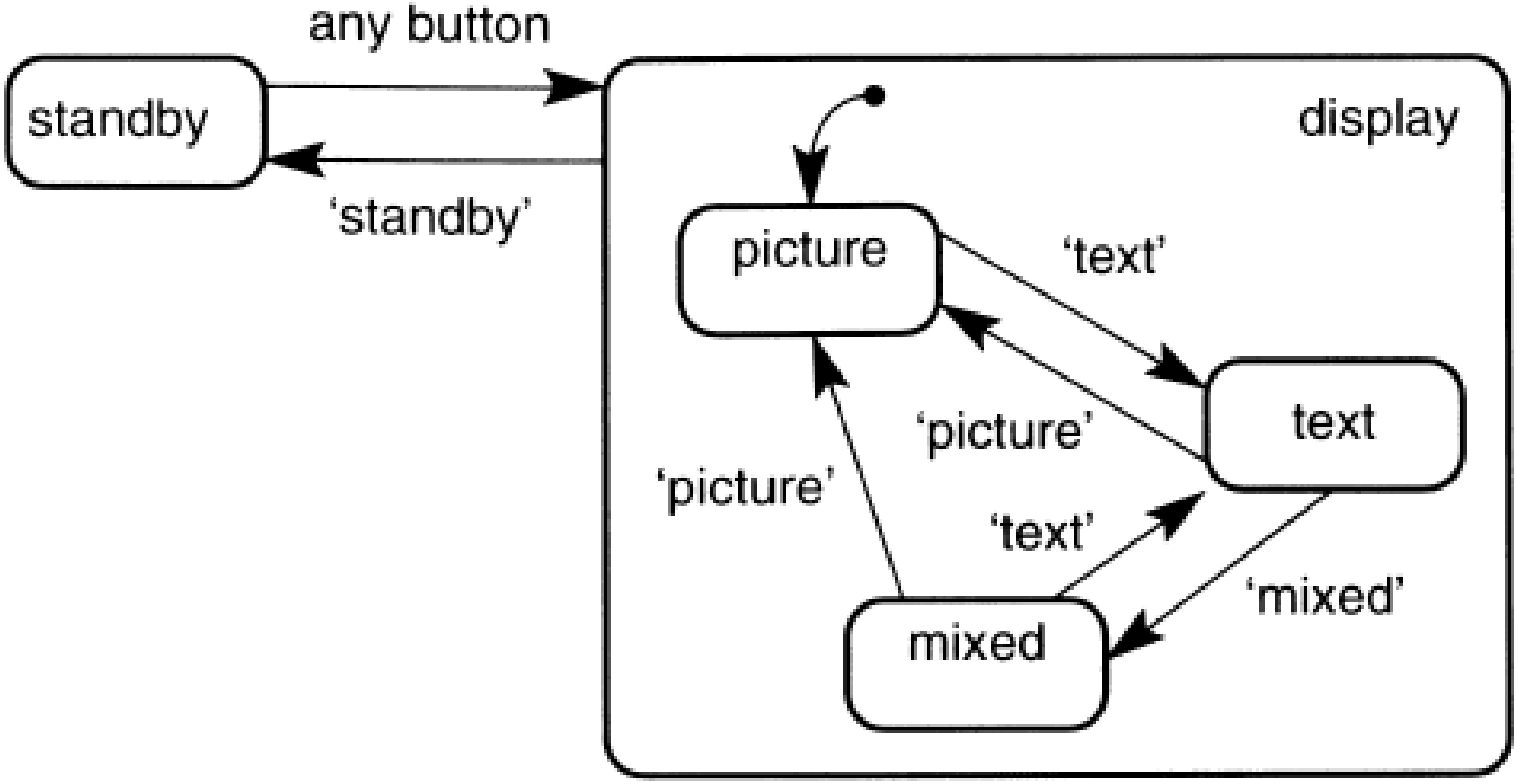
CLEARED FOR TAKEOFF
position on runway;
open throttles

on ground

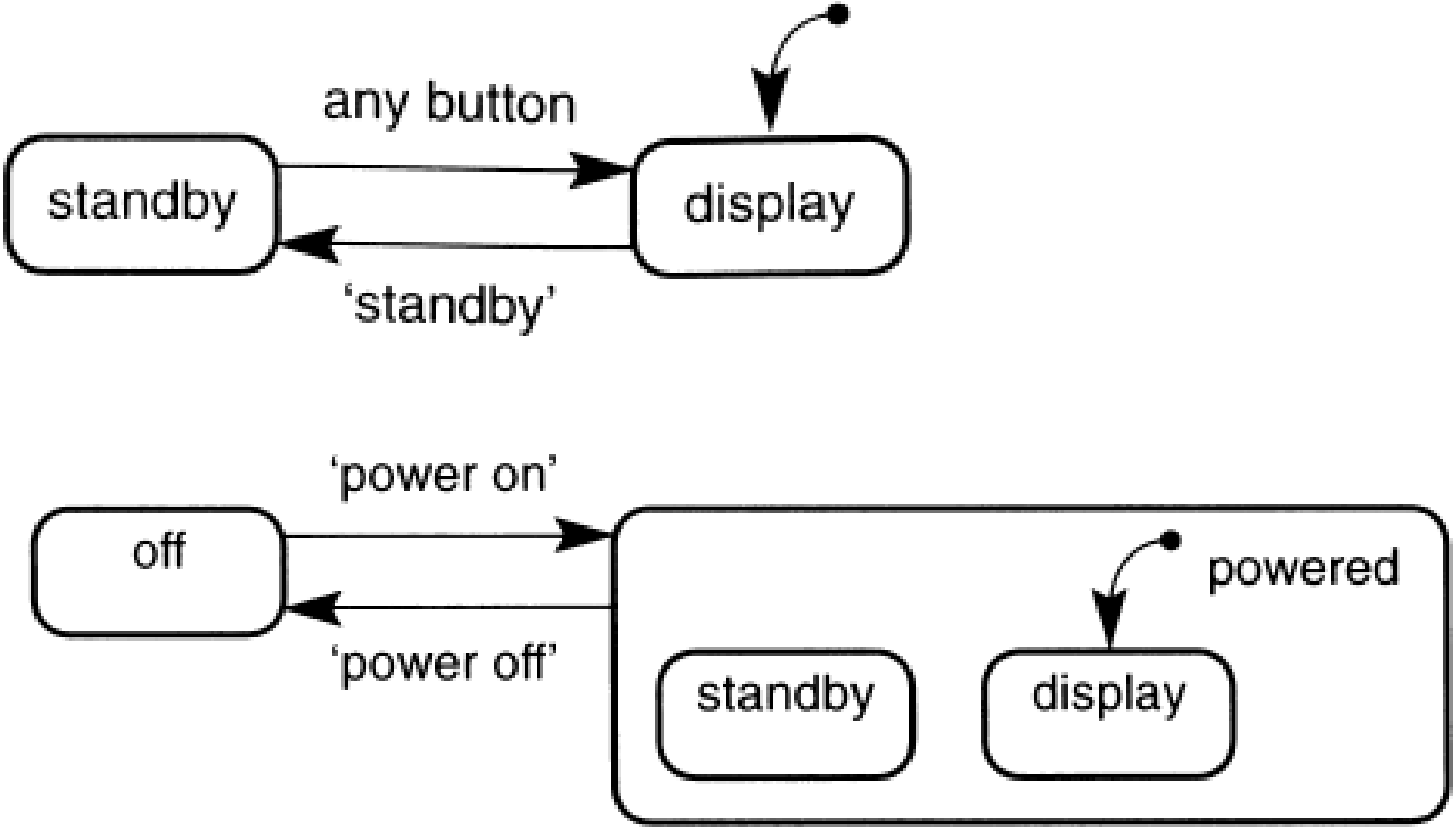
The Statechart



The Statechart



The Statechart

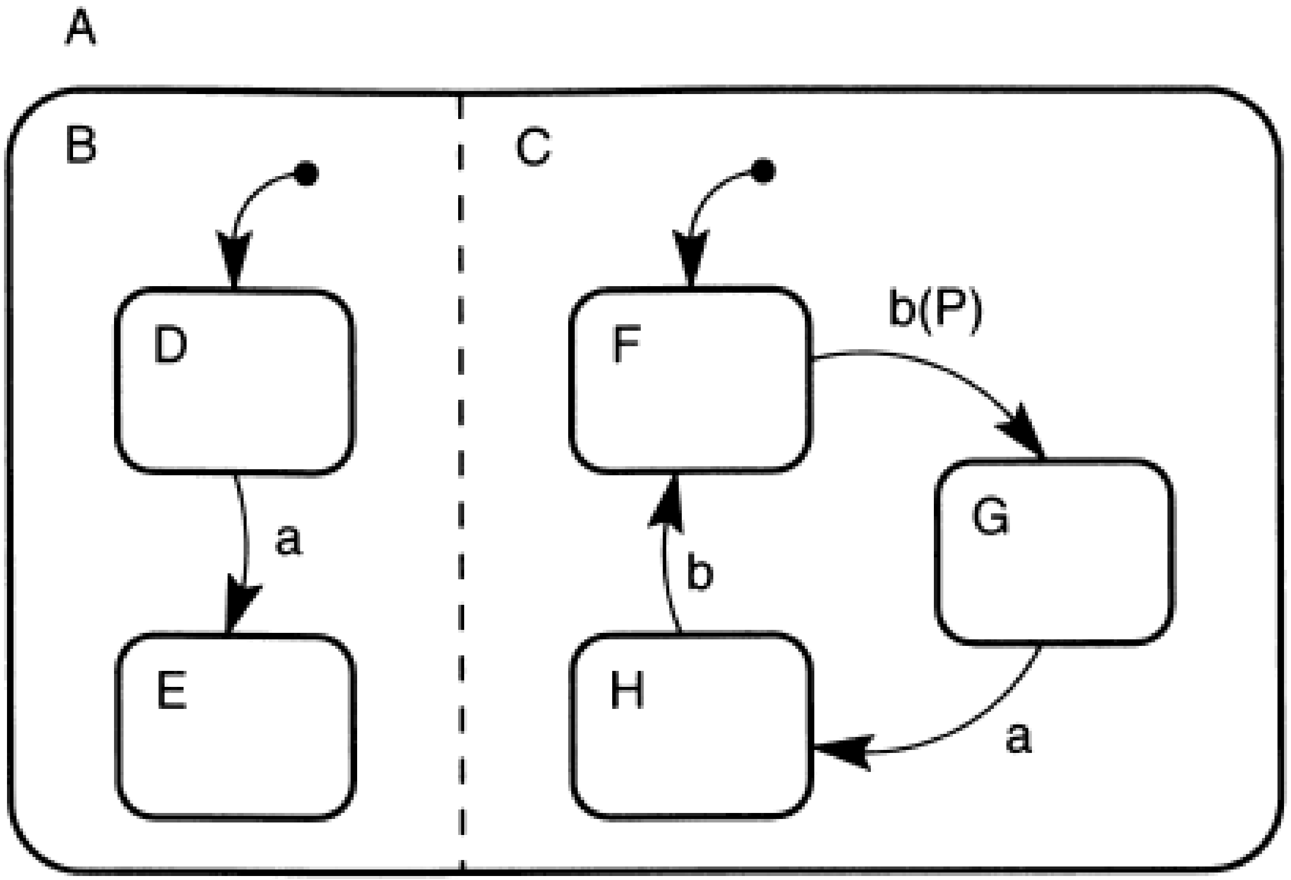


The viewpoint and use of Statechart

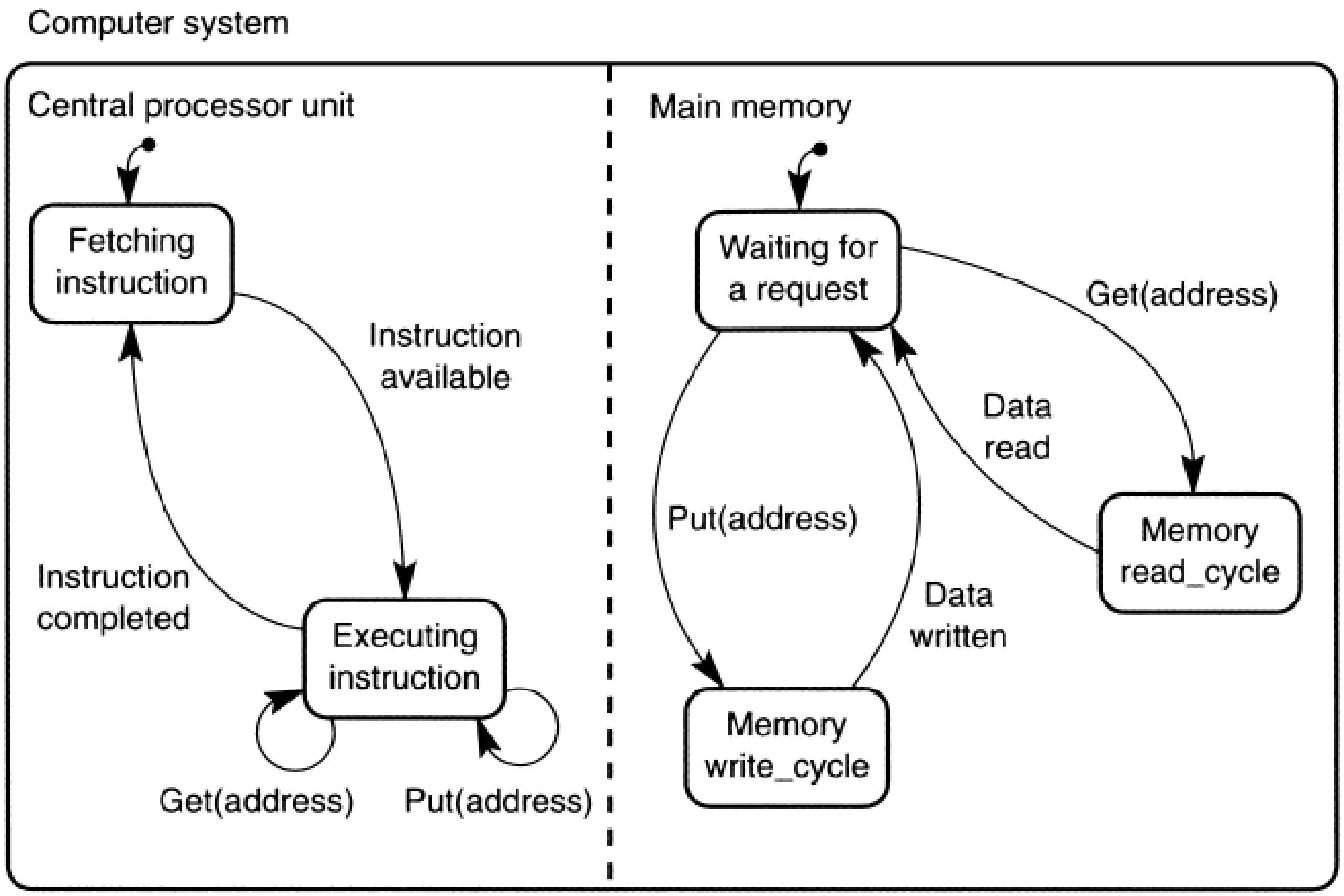
- Provides a behavioural description.
- STD has more detailed descriptions, Statechart deals better with abstractions, defaults, history and scale.
- Used in modelling reactive systems

- Both STD and Statechart support data flow viewpoints.

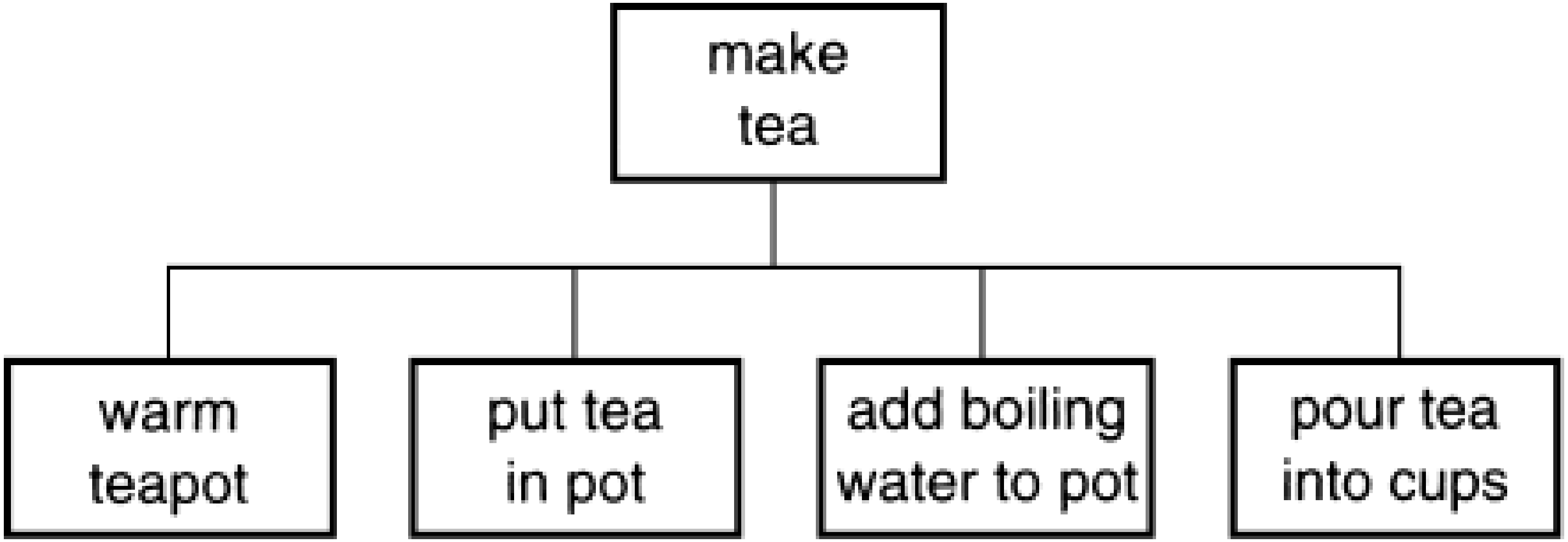
Orthogonality



Orthogonality



Jackson Structure Diagram



make tea

warm teapot

put tea in pot

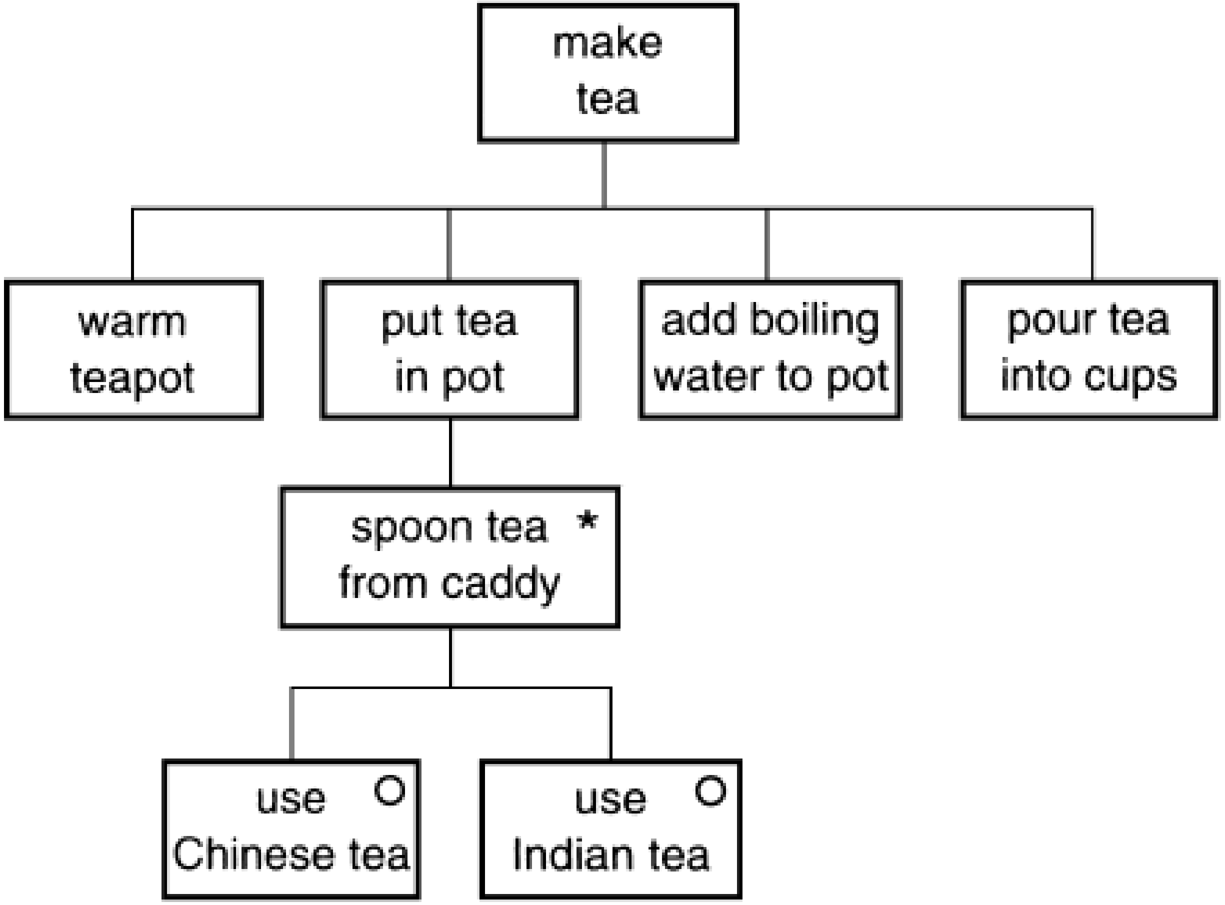
add boiling water to pot

pour tea into cups

Jackson Structure Diagram

- Describes sequential structure with 3 forms:
  - sequence
  - selection
  - iteration
- It can be used to describe:
  - the form of a data structure (data modelling v.p.)
  - sequencing of program actions (functional v.p.)
  - sequencing of states (behavioral v.p.)

Jackson Structure Diagram



make tea

warm teapot

put tea in pot

add boiling water to pot

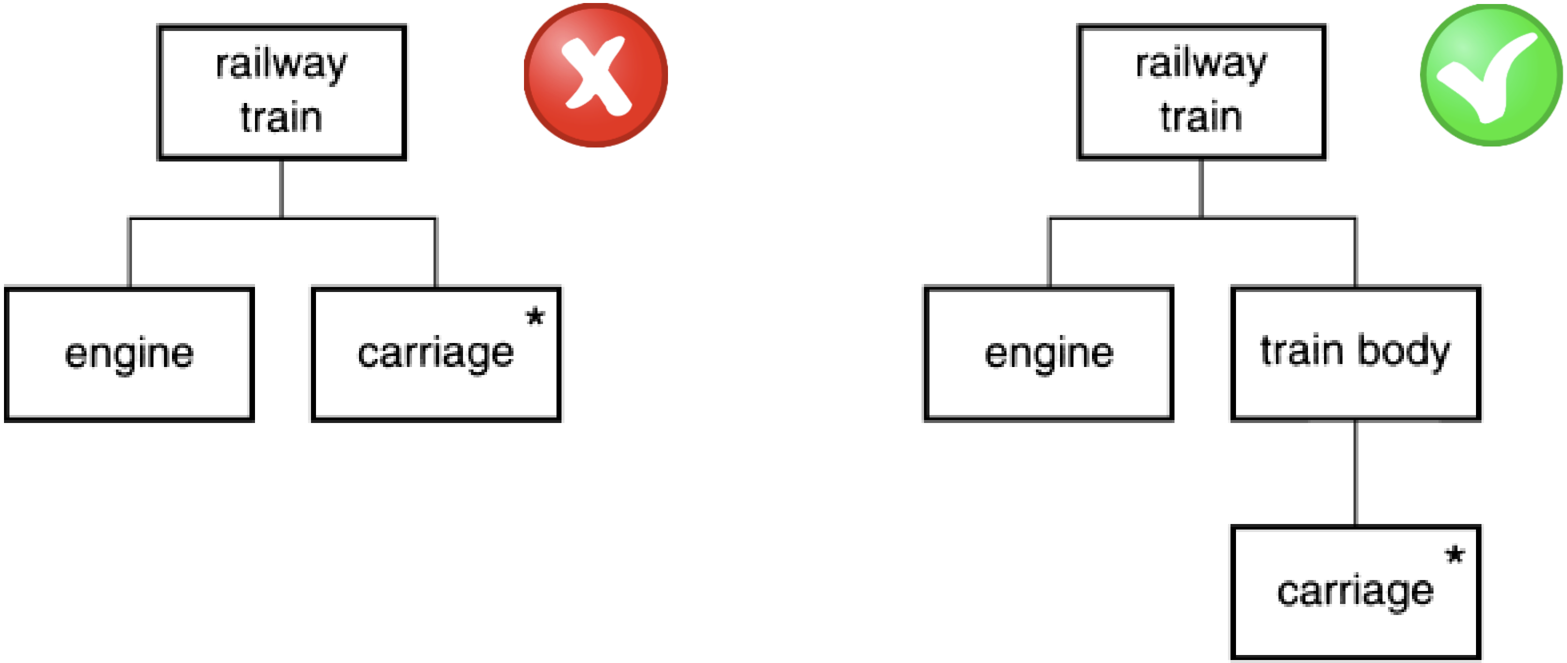
pour tea into cups

spoon tea from caddy *

use Chinese tea ○

use Indian tea ○

Jackson Structure Diagram



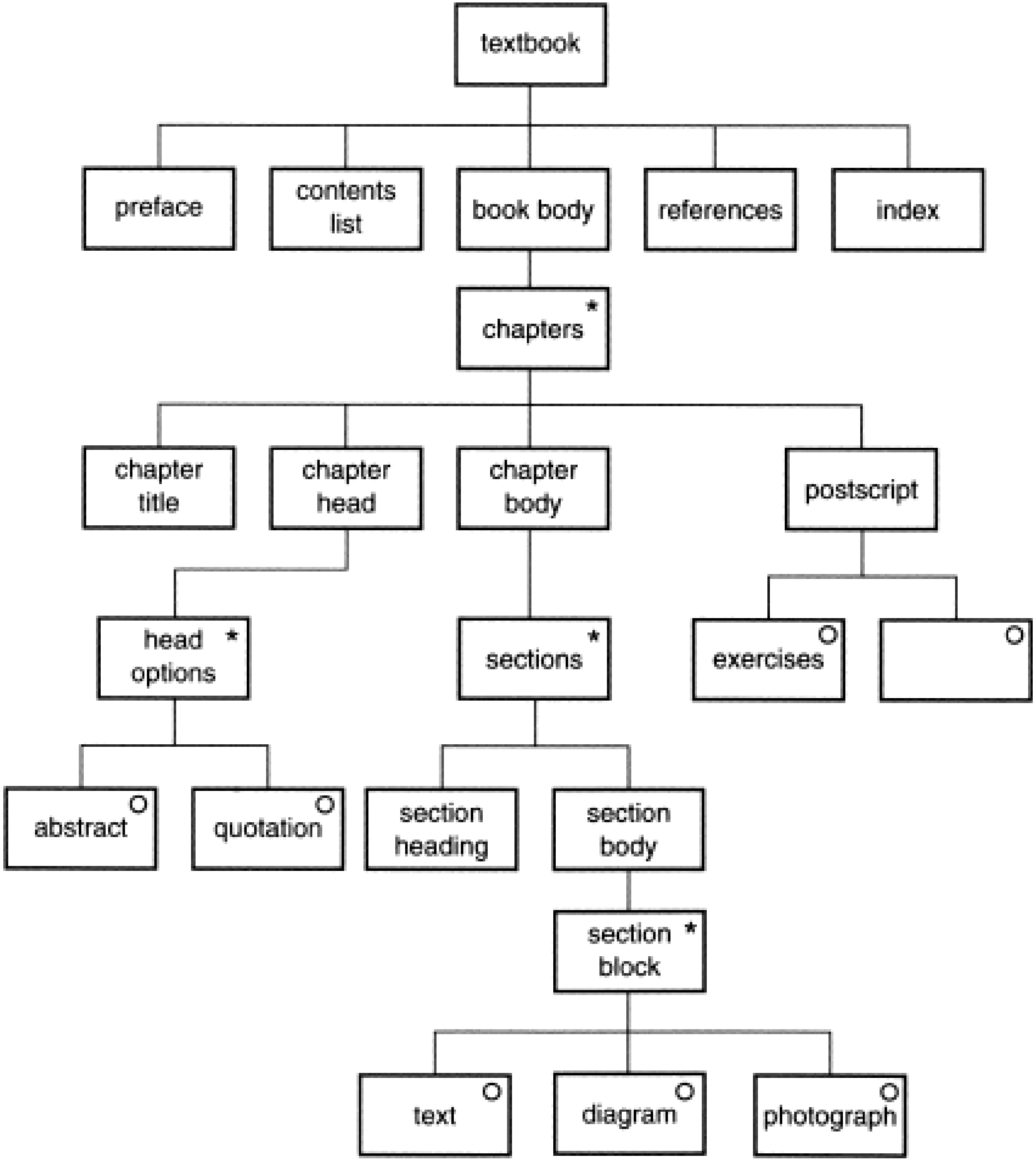
You should **NOT** mix forms within a sequence!

Activity

- Describe the structure of a scientific textbook with Jackson Structure Diagram.

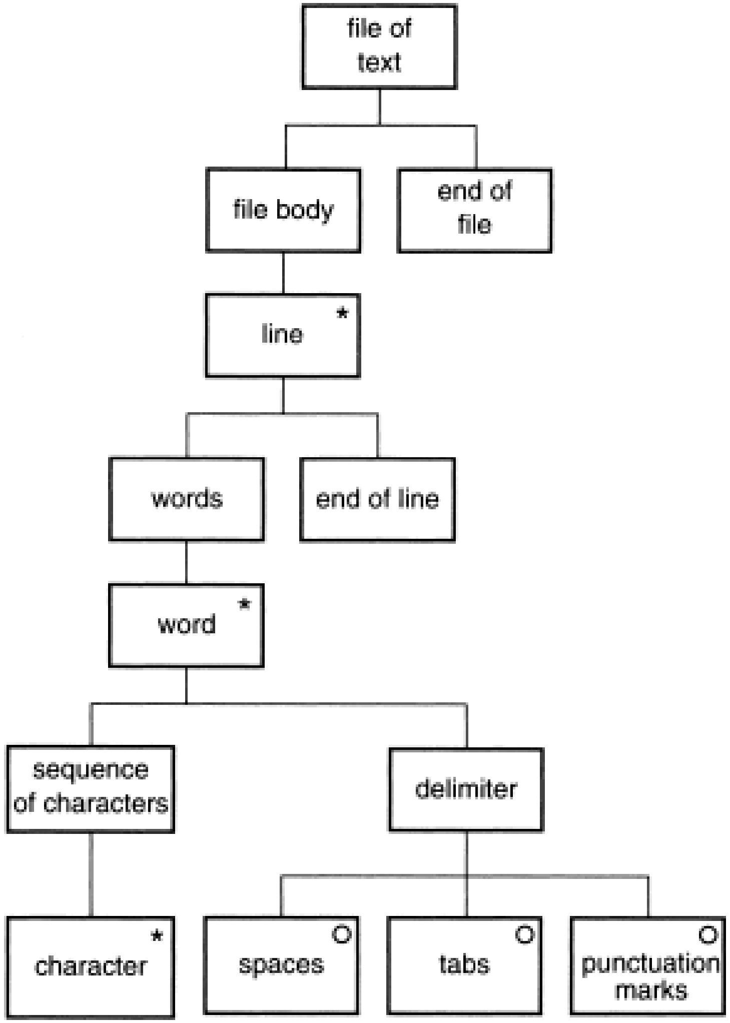
- The book contains a preface, contents, chapters, references and an index.
- Chapters may have exercises at the end
- Chapters can start with an abstract or a quotation, and can contain multiple types of content

Jackson Structure Diagram



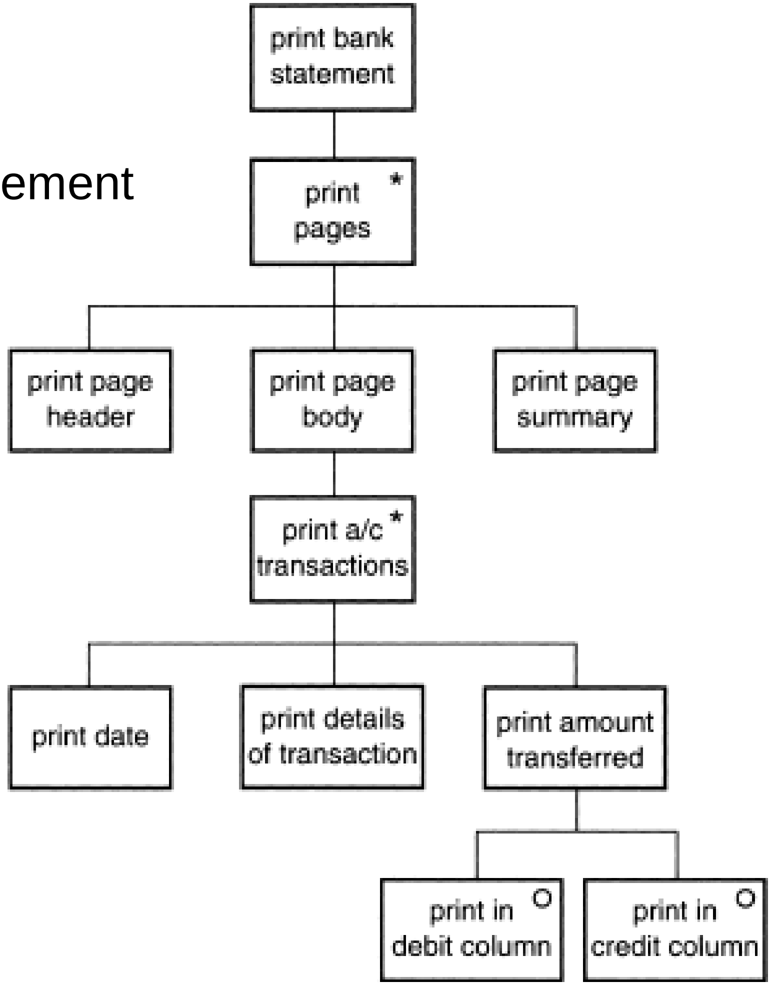
Jackson Structure Diagram

a simple text file

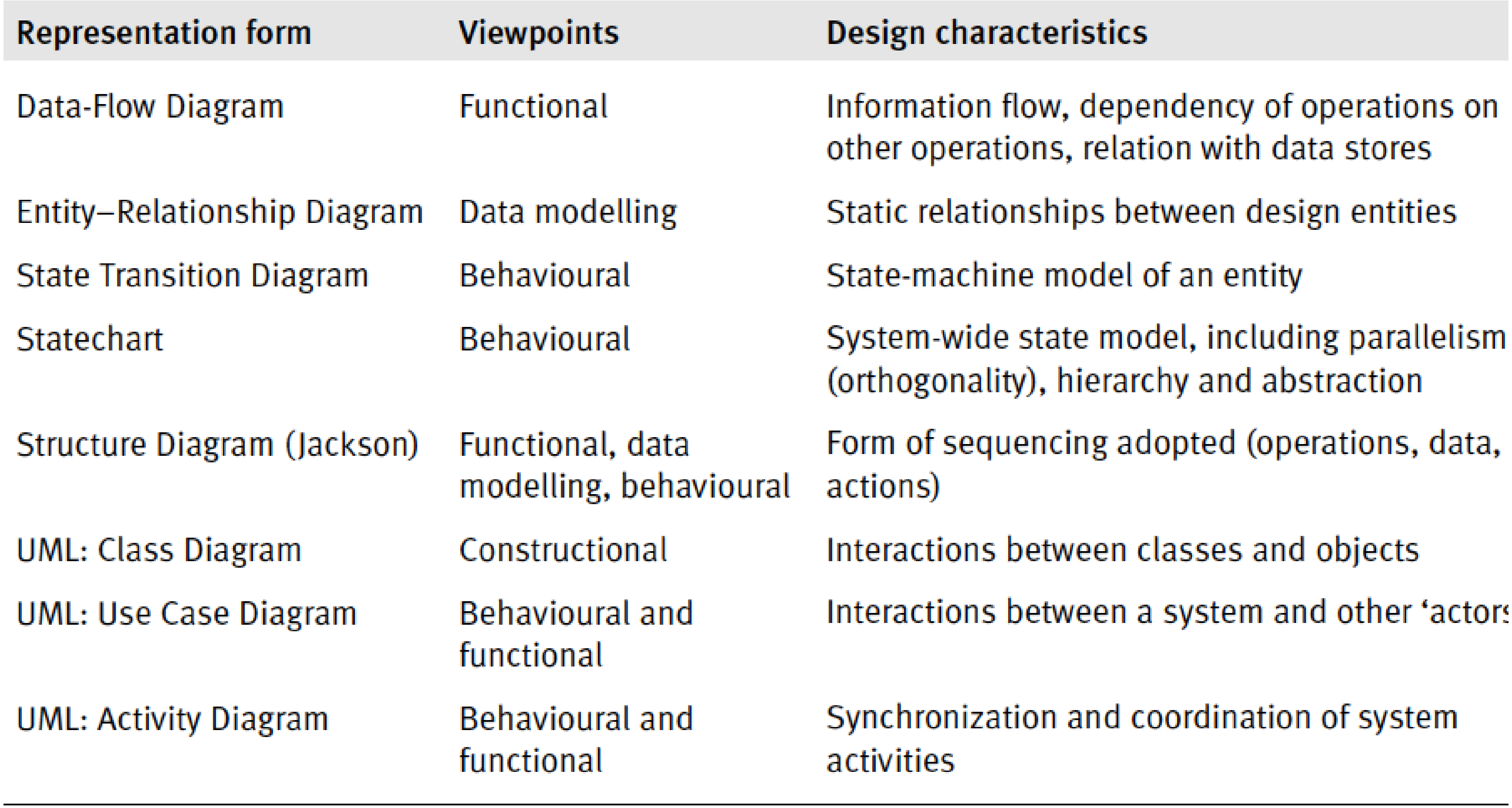


Jackson Structure Diagram

printing a bank statement



Selection of black box forms



Representation form	Viewpoints	Design characteristics
Data-Flow Diagram	Functional	Information flow, dependency of operations on other operations, relation with data stores
Entity–Relationship Diagram	Data modelling	Static relationships between design entities
State Transition Diagram	Behavioural	State-machine model of an entity
Statechart	Behavioural	System-wide state model, including parallelism (orthogonality), hierarchy and abstraction
Structure Diagram (Jackson)	Functional, data modelling, behavioural	Form of sequencing adopted (operations, data, actions)
UML: Class Diagram	Constructional	Interactions between classes and objects
UML: Use Case Diagram	Behavioural and functional	Interactions between a system and other ‘actors
UML: Activity Diagram	Behavioural and functional	Synchronization and coordination of system activities

Summary

- A wide range of notations can be used to support and document the development of a design model
- Examples illustrate the forms used for both white box and black box modelling across the set of viewpoints (functional, behavioural, data modelling and constructional)
- The choice of any particular notation depends upon many factors, including the problem domain, architectural form of the eventual system, and design practices being used.

Questions?