TOPICS

Fundamentals
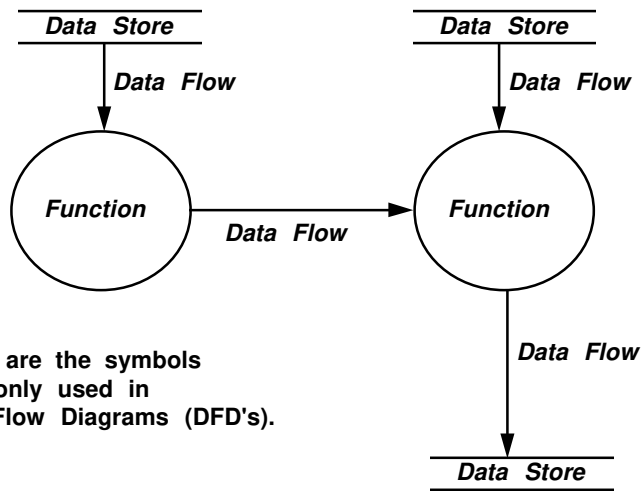
Structured and Object-Oriented Analysis

Formal and Automated Techniques

Data Flow Analysis Methods

- Data Flow Diagrams

- Data Dictionary

Data Flow Diagrams

Data Store

Data Flow

(Function)

Data Flow

Data Store

Data Flow

(Function)

Data Flow

These are the symbols
commonly used in
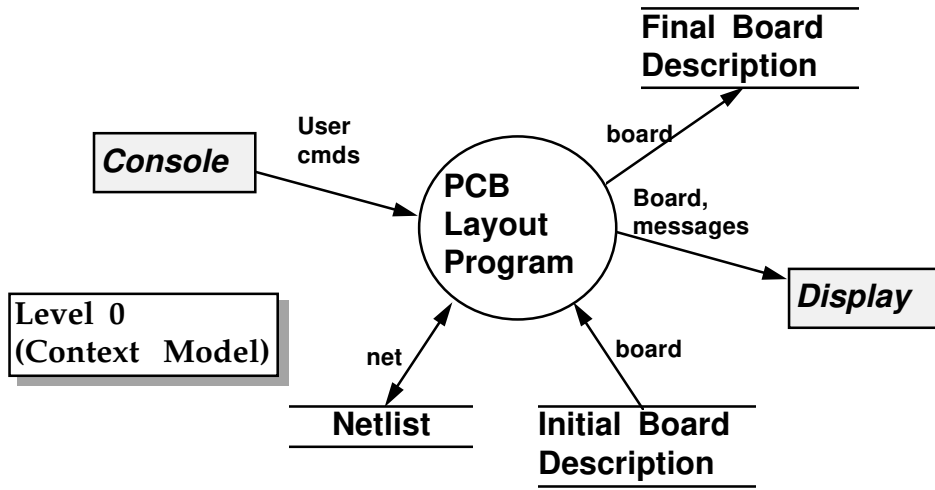Data Flow Diagrams (DFD's).

Data Store

DFD Example

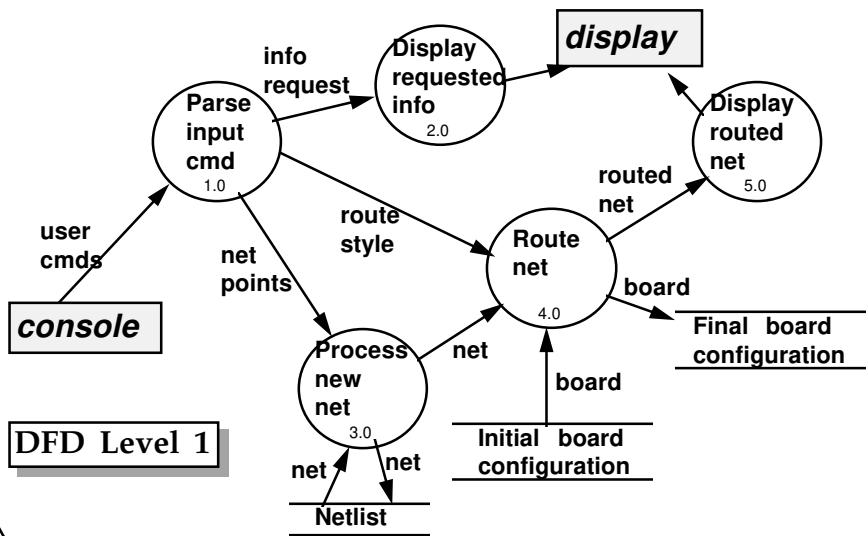## Simple Printed Circuit Board Layout Program

Given two data files: a list of nets and initial board description,

1. Determine and display the best route for interconnecting each net on the board.

2. Permit user to:

    a. add new nets to list

    b. delete nets from list

    c. select any or all nets to be routed

    d. request status info about nets or routed board

    e. define style of routing (steiner points, chain, or tree)

    f. save final routed board in a file

DFD Example, Continued

**Final Board
Description**

Console

User
cmds

PCB
Layout
Program

board

Level 0
(Context   Model)

Board,
messages

Display

net

board

Netlist

**Initial Board
Description**

DFD Example, Continued

DFD Level 1

DFD Example, Continued

1.0 **route style**

board, net

route net using chaining 4.2

board, routed net

routed net

Display

setup net for routing 4.1

board, net

route net using steiner points 4.3

board, routed net

Update board 4.5

net

3.0

board

board, net

Initial board configuration

board, net

route net without steiner points 4.4

board, routed net

board

Final board configuration

DFD Level 2

3B - 7
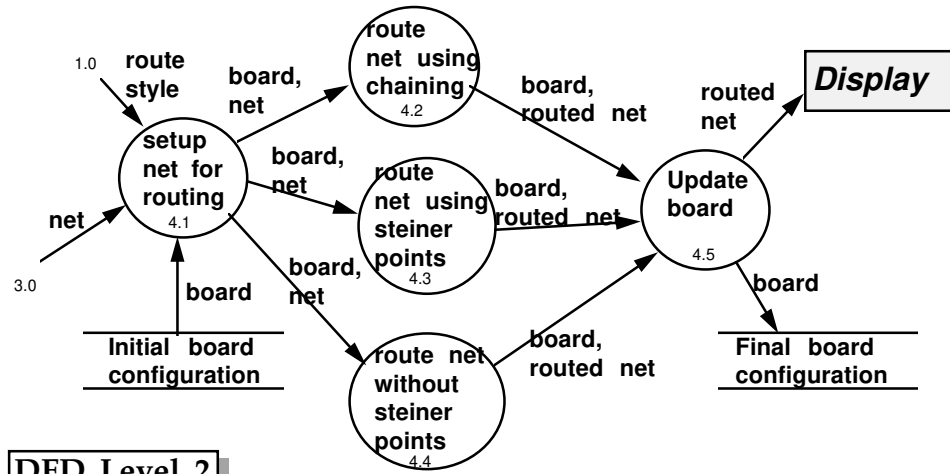
Data Dictionary and Its Content

- **Each class of objects in the system and its attributes**

- **Each singular object (i.e., if placed into a class, the class would have only one instance) and its attributes**

- **Key constants and their attributes**

- **Subprogram parameters and their attributes**

Data Dictionary Entry (Example)

Name: **net**
Alias: **net_graph, point_list**
Used: <u>**process**</u> <u>**in**</u>   <u>**out**</u>   <u>**file**</u> <u>**buffer**</u> <u>**external**</u>
      4.1   **3.0 4.2,4.3,4.4**
      4.2   **4.1**
      4.3   **4.1**
      4.4   **4.1**
Description: **List of no more than 20 points (x,y) where x and y are vertical and horizontal grid locations on the board. x and y are 16-bit unsigned integer values each greater than 0 and less than the max size of the board.**
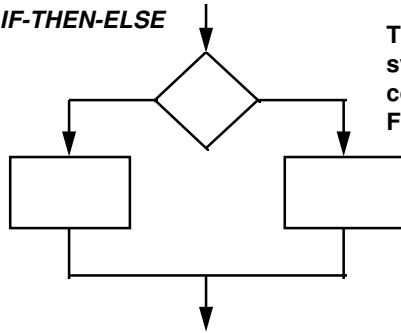*Supplementary Information:* **-- none --**

Functional   Analysis   Methods

- **Function   Diagrams**
- **State Transition Diagrams (STD's)**
- **Entity-Relationship   Diagrams   (ERD's)**
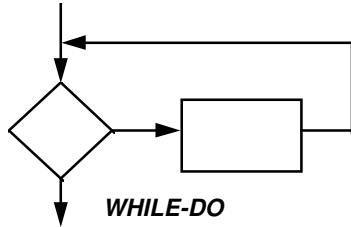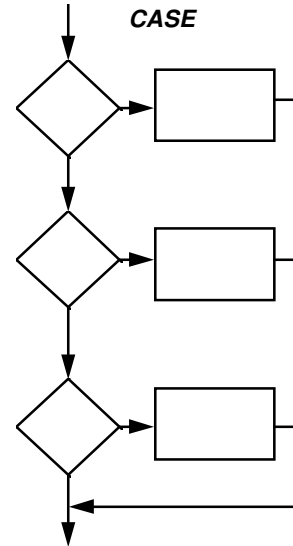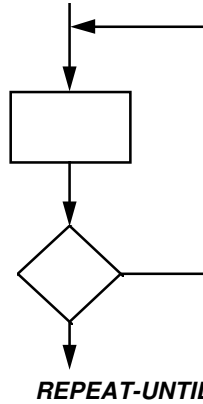
Function Diagrams

IF-THEN-ELSE

These are the symbol combinations commonly used in Function Diagrams.

CASE

WHILE-DO

REPEAT-UNTIL

Function Diagrams - Example

| Open File | | Get Line | | Done? | Yes | Close File |

```
Open ──────► Get ──────► ◇ Done? ──Yes──► Close
File         Line          │              File
  ▲                        │
  │                       No
  │                        │
  │                        ▼
  │                      Put
  └──────────────────── Line
```

Open File → Get Line → Done? — Yes → Close File

Done? — No → Put Line → (loop back to Get Line)
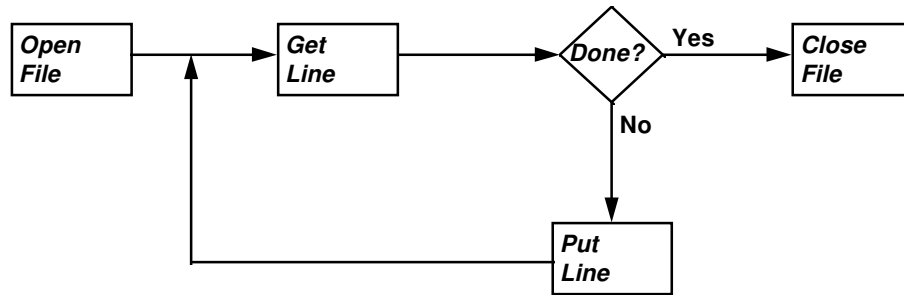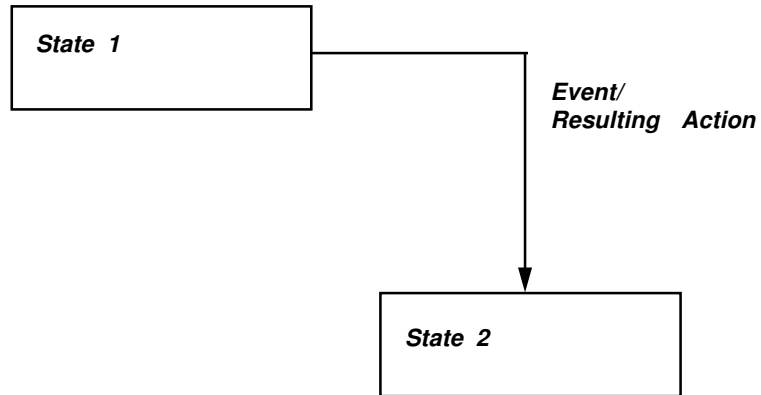
Behavioral Modeling

- **Helpful for control-dominated systems**

- **State Transition Diagrams**

    - **Like Finite State machines**

    - **Depicts states and events causing change of state**

    - **Depicts actions to be taken when events received**

3B - 14

Software Engineering

State Transition Diagrams

```
State  1
```

Event/
Resulting   Action

```
State  2
```

These are the symbols
commonly used in
State Transition Diagrams (STD's).

3B - 14

State Transition Diagrams - Example

```
┌──────────────────────┐                    ┌──────────────────────┐
│ Initialize System and│      Done          │  Read Line from File │◄───┐
│ Open  File           │ ─────────────────► │                      │    │
└──────────────────────┘                    └──────────────────────┘    │
                                                      │ New  Line  Read  │
 Open  Failed/          End  of File                  ▼                  │
 File  Not                                  ┌──────────────────────┐    │
 Found  Message                             │  Format  Line        │    │
 Is  Displayed                              │                      │    │
                                            └──────────────────────┘    │
                                                      │ Done             │
                                                      ▼                  │
┌──────────────────────┐                    ┌──────────────────────┐    │
│ Close File (if open) │◄───────────────────│ Write Line to Console│────┘
│ and  Exit            │  Write  Failed/    │                      │ Write
└──────────────────────┘  Error  Message    └──────────────────────┘ Successful
                          Displayed
```
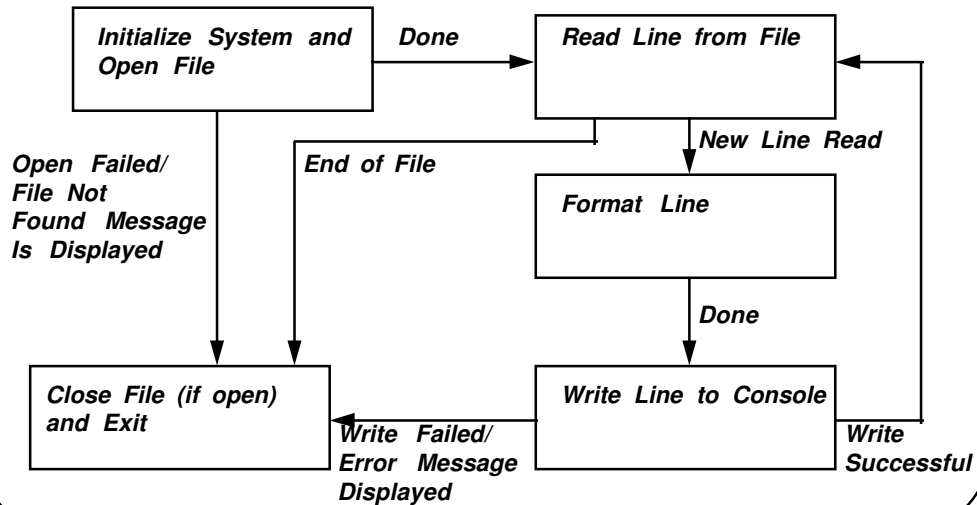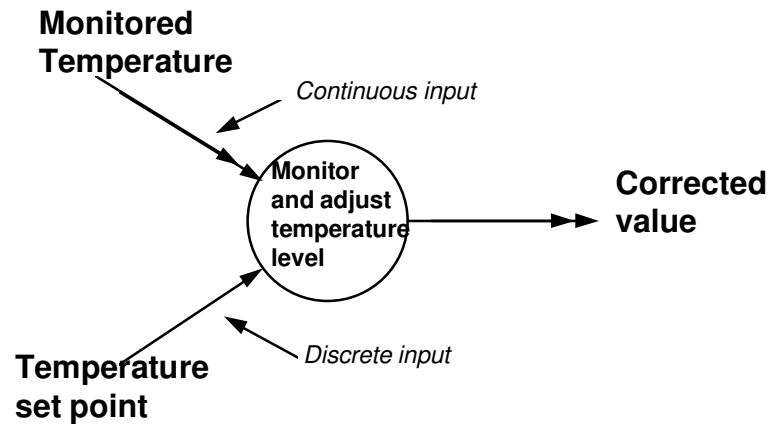
SRA for Real Time Systems

- *Real Time Systems:*

    1. Time dependent

    2. Control oriented

    3. Driven by events more than data

    4. Some activities execute asynchronously

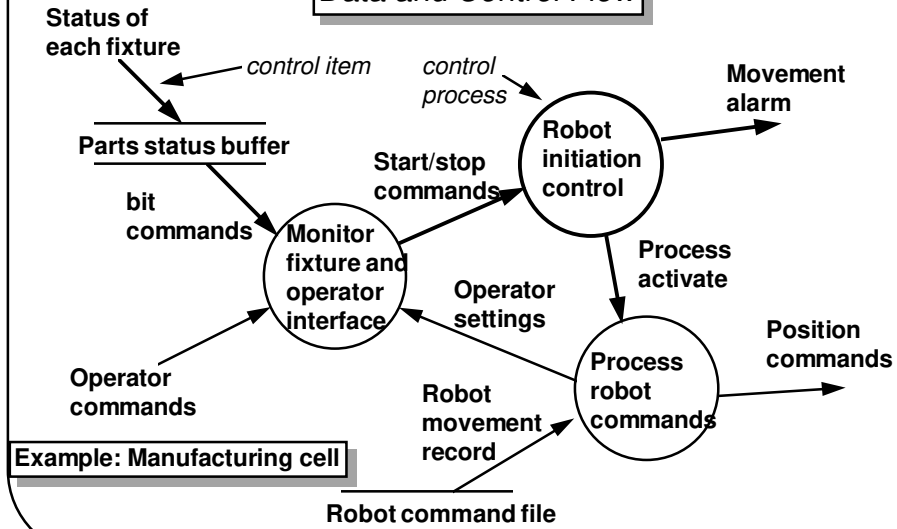- Use <u>control</u> flow models to specify such systems
- Approach: Extend DFD model
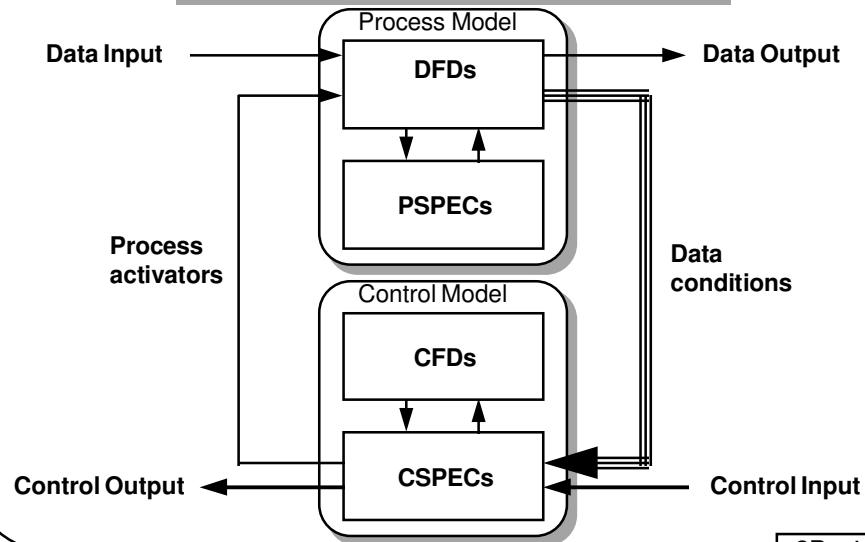
Ward-Mellor Extensions

Time Continuous Data Flows

Monitored Temperature

Continuous input

Monitor and adjust temperature level

Corrected value

Temperature set point

Discrete input

Ward-Mellor Extensions, Continued

Data and Control Flow

Status of each fixture

control item

control process

Movement alarm

Parts status buffer

Robot initiation control

Start/stop commands

bit commands

Monitor fixture and operator interface

Process activate

Operator settings

Operator commands

Position commands

Robot movement record

Process robot commands

Example: Manufacturing cell

Robot command file

3B - 18

Hatley-Pirbhai Extensions

Combined Data Flowand Control Flow

Process Model

Data Input → **DFDs** → **Data Output**

PSPECs

Process activators

Data conditions

Control Model

CFDs

Control Output ← **CSPECs** ← **Control Input**

3B - 19

Hatley-Pirbhai Extensions, Continued

Example: Office Photocopier Software

DFD

Operator commands and data

Copy info

Manage copying

Copy status

Produce user displays

Displays

Read operator input

Problem code

Problem type

Reload requirements

Reload status

Perform problem diagnosis

Reload paper

3B - 20
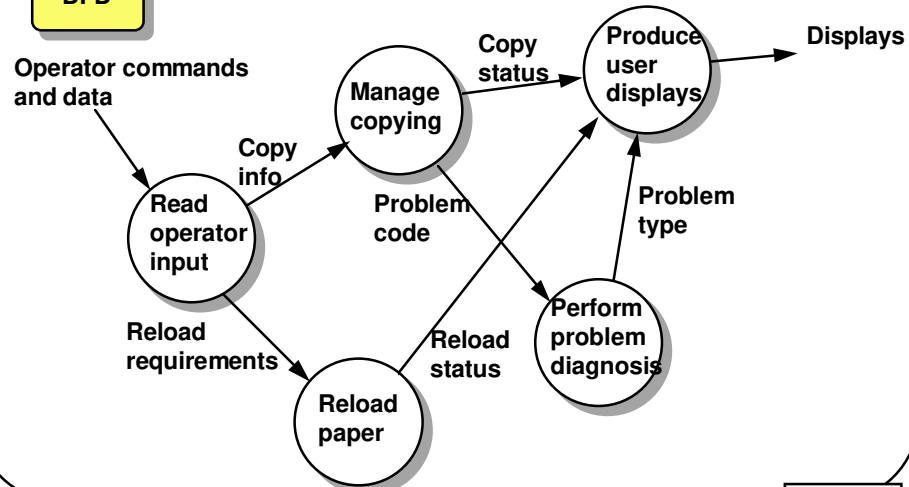
Hatley-Pirbhai Extensions, Continued

CFD

Example: Office Photocopier Software

Paper feed status (jammed, empty, etc.)

Alarm

Start/ stop

Manage copying

Produce user displays

Read operator input

Perform problem diagnosis

Reload paper

Full

Repro fault

3B - 21
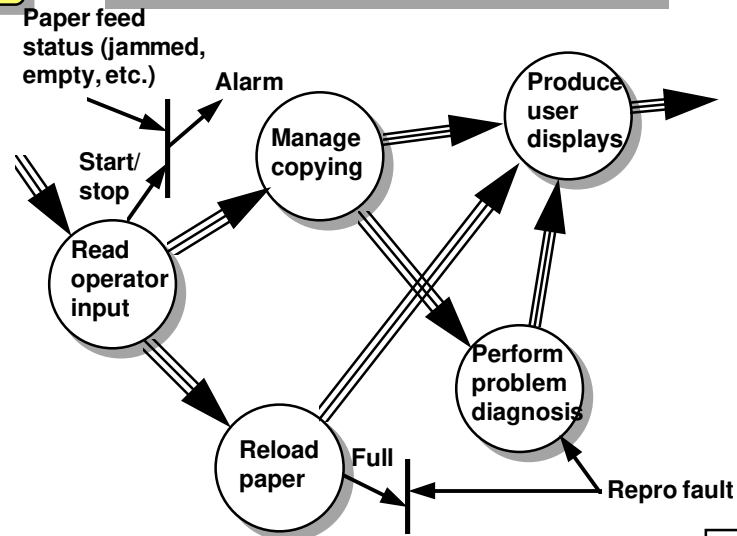
Hatley-Pirbhai Extensions, Continued

Example: Office Photocopier Software

PSPEC

Read Operator Input:

if op_in = paper11

    then set form=11 inches;

if op_in = paper14

    then set form=14 inches;

if op-in = color

    then set style=colortype;

     .

     .

     .

CSPEC

Alarm Condition:

if start && (feed_status = NOT

   ok) then set alarm (status);

    .

    .

    .

3B - 22

An Alternative: SADT [1]

**Structured Analysis and Design Technique
(also known as IDEF 0)**

nical notation

ms and **datagrams** that omminicate relations of information
nd control) and function within software

s project control guidelines for applying methodology

Control

Input → | **entity** | → **Output**

Mechanism
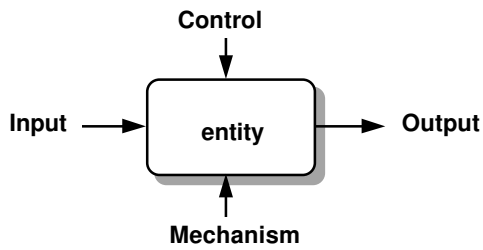
SADT, Continued

Actigrams and Datagrams

Actigram

Report request

Records → | Report status | → Report
→ Records

Modify status

Datagram

Generate records

| Records | → Report status

Maintain and update
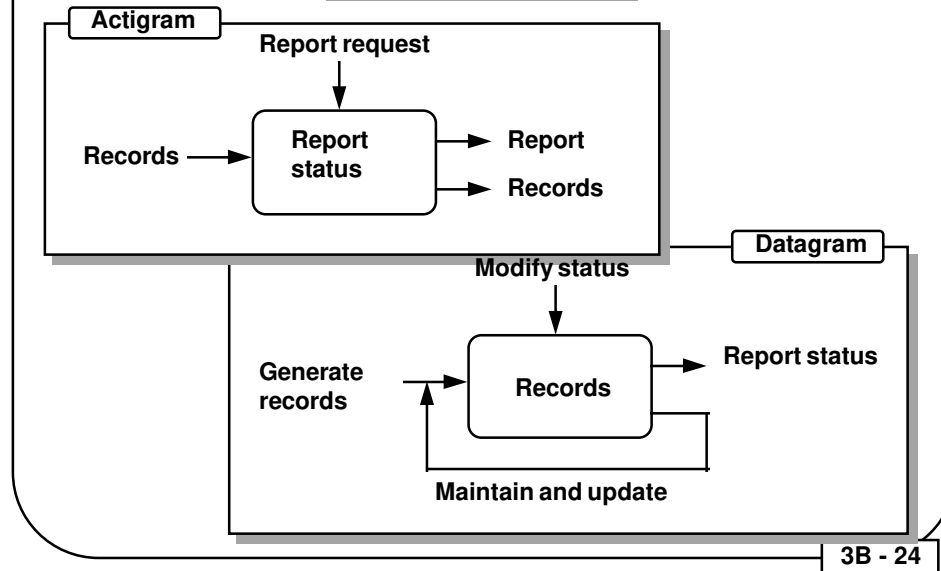
SADT, Continued

Example: Spelling Checker

**Request for more
words to check**

**Words
to check**

**Text to
check**

**Read
input**

**Search
for errors**

Dictionary

**Report
errors**

Report

3B - 25
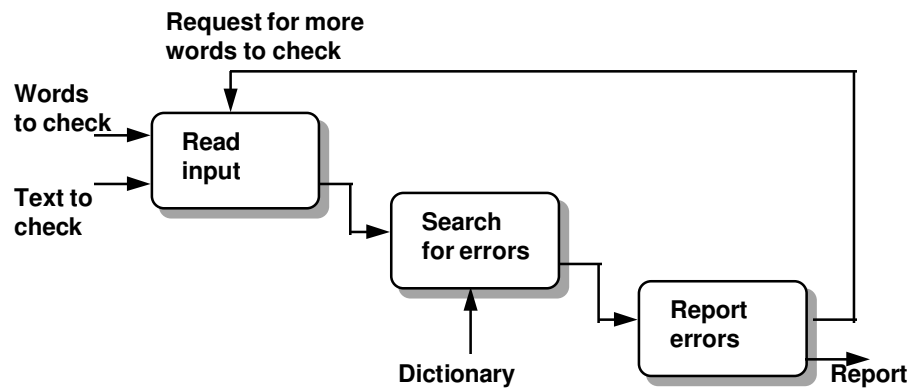
OOA: Object Oriented Analysis

- **Basic concepts**
- **How to identify objects**
  - **Identifying objects**
  - **Specifying attributes**
  - **Defining Operations**
  - **Communication between objects**
- **OOA modeling**
  - **Classification and assembly structures**
  - **Defining subjects**
  - **Instance connections and message paths**
  - **Prototyping**
- **Data Modeling**
  - **Data objects, attributes and relationships**
  - **E-R diagrams**

Basic Concepts

Object Oriented Development Process

Given a clear and complete statement of problem definition:

Identify Objects

Identify Structures

Define Subjects

Define Attributes and instance connections

Define Operations and Message Connections

Coad, P., and E. Yourdon, *Object Oriented Analysis*, Prentice-Hall, 1990.

3B - 27

Basic Concepts, Continued

Data

Procedures
- Private
- Public

Object

Public procedures

Private procedure
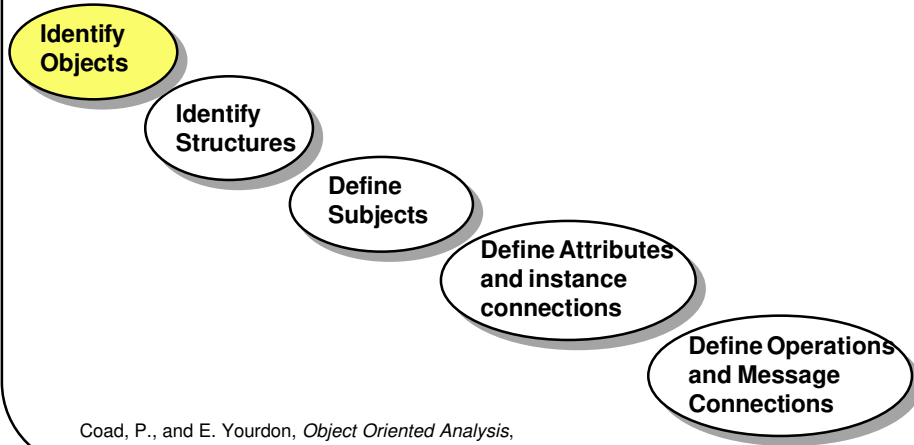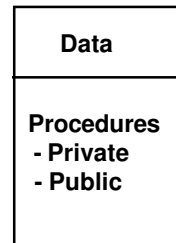
Booch Diagram
of an object

Objects are specific instances of classes (i.e., templates)

Basic Concepts, Continued

Objects are specific instances of classes (i.e., templates)

Data Declaration (i.e., attributes)

Procedure Declarations
- Private
- Public

Class

Object

Objects inherit all attributes and operations of the class
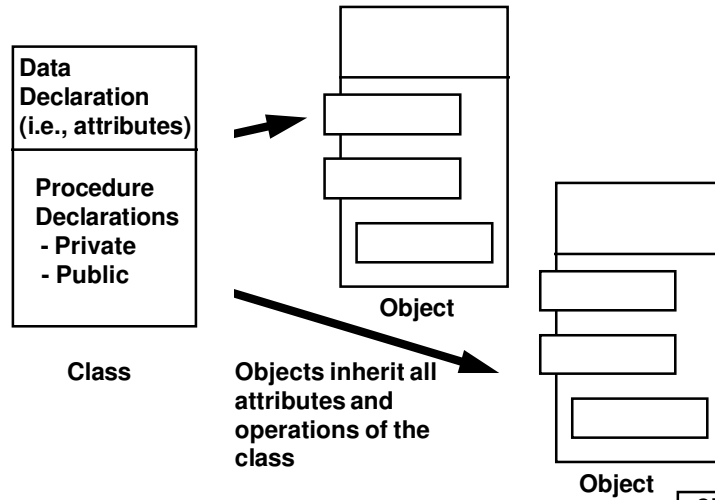
Object

Basic Concepts, Continued

Class/object Example

Chair Object

Cost Dimensions Weight Location

Furniture Class

Cost Dimensions Weight Location
Buy Sell Weigh Move

Buy

Sell

Weigh

Move

Table Object

Cost Dimensions Weight Location

Buy

Sell

Weigh

Move

3B - 30
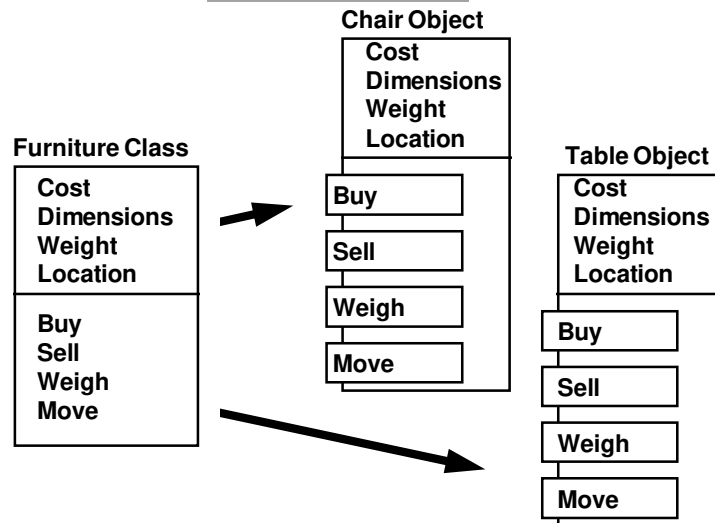
Basic Concepts, Continued

- **Encapsulation -   All class information is contained under one name which can be reused as one specification or program component.**

- **Inheritance - Objects and derived classes inherit all attributes and operations from their class descriptions.**

- **Polymorphism -   Derived classes can add, delete, and redefine inherited attributes and operations.**

- **Messages - Procedures in separate objects communicate (i.e., call and return) via messages.**

How to Identify Objects

Identifying Objects

Object Name

Attributes

Operations
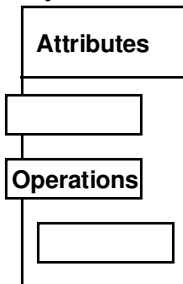
Potential Objects - examples

- **External entities - devices, people**

- **Things - reports, displays, signals**
- **Occurances or events - interrupts**
- **Roles - manager, engineer**
- **Organizational units - division, group**
- **Places - shop floor, tail section**
- **Structures - sensors, computers**

Identifying Objects - Example

Find the potential objects in the following narrative:

Safehome software enables the homeowner to configure the security system when it is installed, monitors all sensors connected to the security system, and interacts with the homeowner through a key pad and function keys contained in the *SafeHome* control panel.

During installation, the *SafeHome* control panel is used to "program" and configure the system. Each sensor is assigned a number and type, a master password is programmed for arming and disarming the system, and telephone number(s) is (are) input for dialing when a sensor event occurs.

When a sensor event is sensed by the software, it rings an audible alarm attached to the system. After a delay time that is specified by the homeowner during sysem configuration activities, the software dials a telephone number of a monitoring service, provides information about the location, and reports the nature of the event that has been detected. The number will be redialed every 20 secondss until telphone connection is abtained. .....

Potential Object/class   Classification

Identifying Objects - Example

Selection Criteria for classes and objects:

1. **Retained information - information that must be remembered for system to function.**

2. **Needed services - operation are needed to change values of attributes.**

3. **Multiple attributes - focus on "major" information. Single or minor attributes can be collected together in single object.**

4. **Common attributes - attributes which apply to all occurances of the object.**

5. **Common operations - operations which apply to all occurances of the object.**

6. **Essential requirements - external entities that produce or consume information that is essential to system operation.**

Potential Object/Class   Applicable Criteria

How to Identify Objects

Specifying Attributes

1. Scan the problem definition and select those things that belong to an object.

2. For each object, ask "what data items (composite or elementary) fully define this object in the context of the problem?

3. For example, using the *SafeHome* system object:

> sensor_info = sensor_type + sensor_number + alarm_threshold
>
> alarm_response = delay_time + telephone_number + alarm_type
>
> activate/deactivate_info = master_password + tries_allowed + temp_password
>
> id_info = system_ID + verification_phone_no. + system_status

How to Identify Objects

Defining Operations

Operations are of three types:

- **Manipulation - add, delete, reformat, select, initialize**
- **Computation - equations, transformations**
- **Monitoring - occurance of a controlling event**

To derive a set of operations for each object:

1. **Scan the problem definition and grammatically parse it for verbs to be candidate operations that belong to each object.**

2. **Try defining the candidate operations for objects defining the SafeHome system (use description in prior slide)**

How to Identify Objects

Interobject Communication

During requirements definition, eplicite messages need not be known. Only general object interaction should be defined.
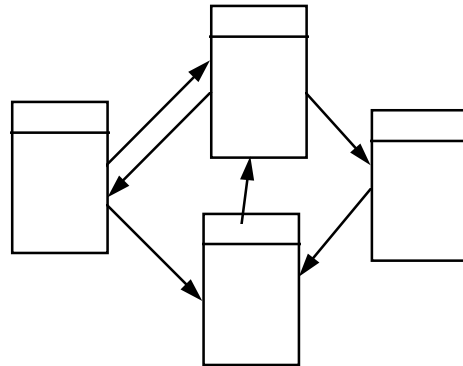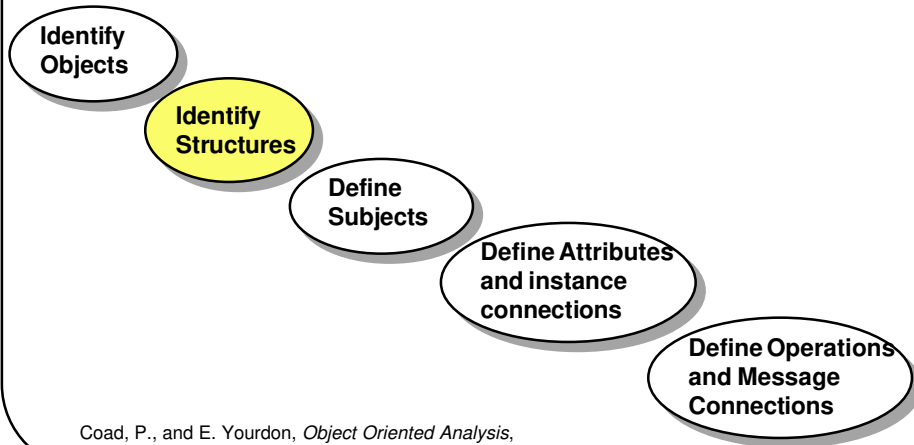

Note: Messages initiated by procedures in objects, and are sent to precedures in objects.

msg: (dest, op, args)

3B - 37

Object-Oriented Development Process

Given a clear and complete statement of problem definition:

Identify Objects

Identify Structures

Define Subjects

Define Attributes and instance connections
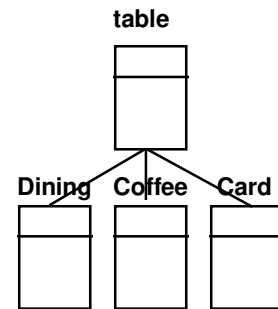
Define Operations and Message Connections

Coad, P., and E. Yourdon, *Object Oriented Analysis*, Prentice-Hall, 1990.
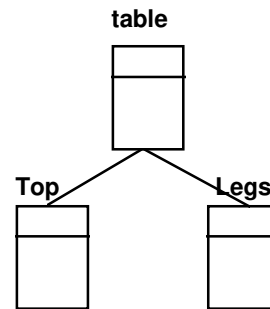
3B - 38

OOA Modeling

Classification and Assembly Structures

Once objects have been defined, structure groups of them into <u>classification</u> trees or <u>assembly</u> trees:
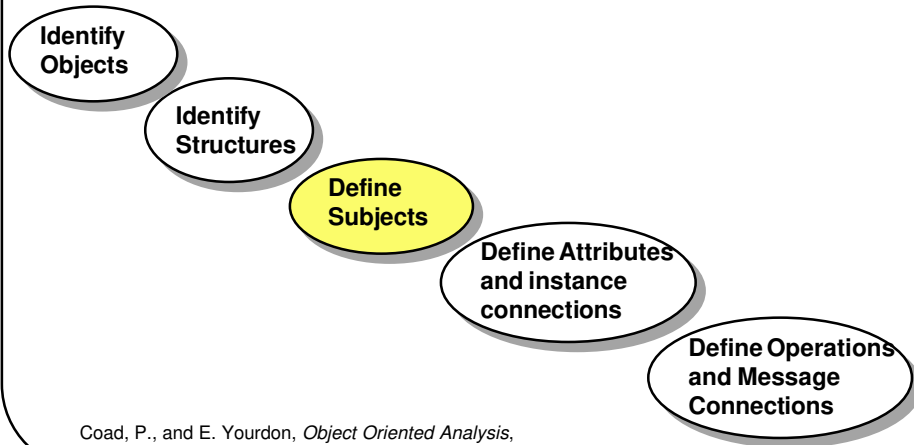

Classification Structure   **Assembly Structure**

Object-Oriented Development Process

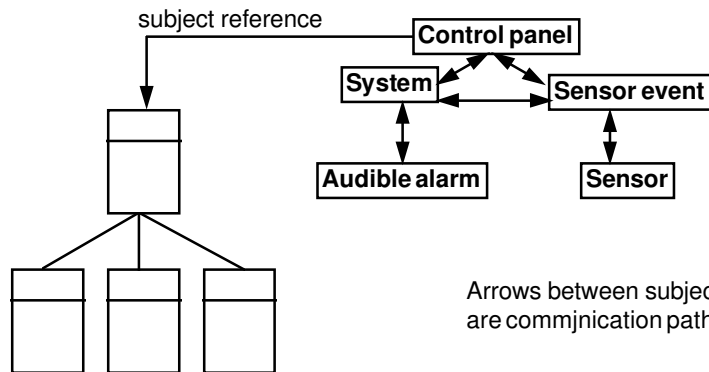**Given a clear and complete statement of problem definition:**

Identify
Objects

Identify
Structures

Define
Subjects

Define Attributes
and instance
connections

Define Operations
and Message
Connections

Coad, P., and E. Yourdon, *Object Oriented Analysis*,
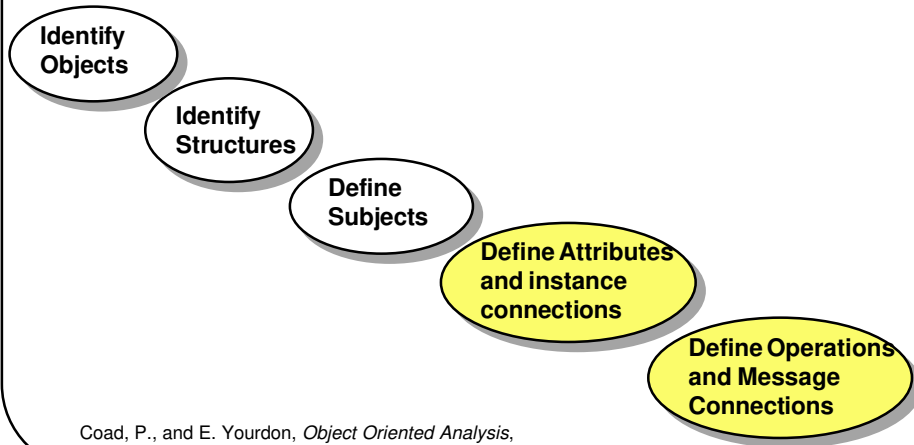Prentice-Hall, 1990.

3B - 40

OOA Modeling

Defining Subjects

For large OOA models with hundreds of objects and dozens of structures, organize the structures in to subjects which can be referenced by a single name or ID.

subject reference

Control panel

System   Sensor event

Audible alarm   Sensor

Arrows between subjects
are commjnication paths.

3B - 41

Object-Oriented Development Process

Given a clear and complete statement of problem definition:

Identify Objects

Identify Structures

Define Subjects

Define Attributes and instance connections

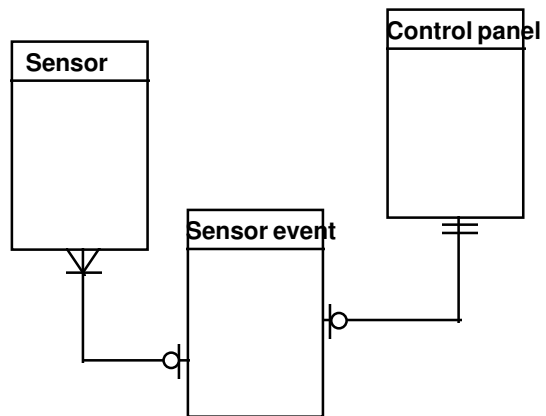**Define Operations and Message Connections**

Coad, P., and E. Yourdon, *Object Oriented Analysis*, Prentice-Hall, 1990.

3B - 42

OOA Modeling

Instance Connections and Message Paths

**Analyist should define specific relationships
between objects:**

Sensor

Control panel

Sensor event

Define:

○   **zero**

| **one**

〈   **many**

Thus:

──○   **zero or one**

──╫──   **exactly one**

──K   **one or more**

──○K   **zero or more**

3B - 43

OOA Modeling

Prototyping

OOA can lead to very effective prototyping
  techniques

- Reuse defined, coded, and tested objects

- Establish library of quality objects and save
  analysis info as well as code and tested
  objects

- Use existing object specifications in the
  development of new products.