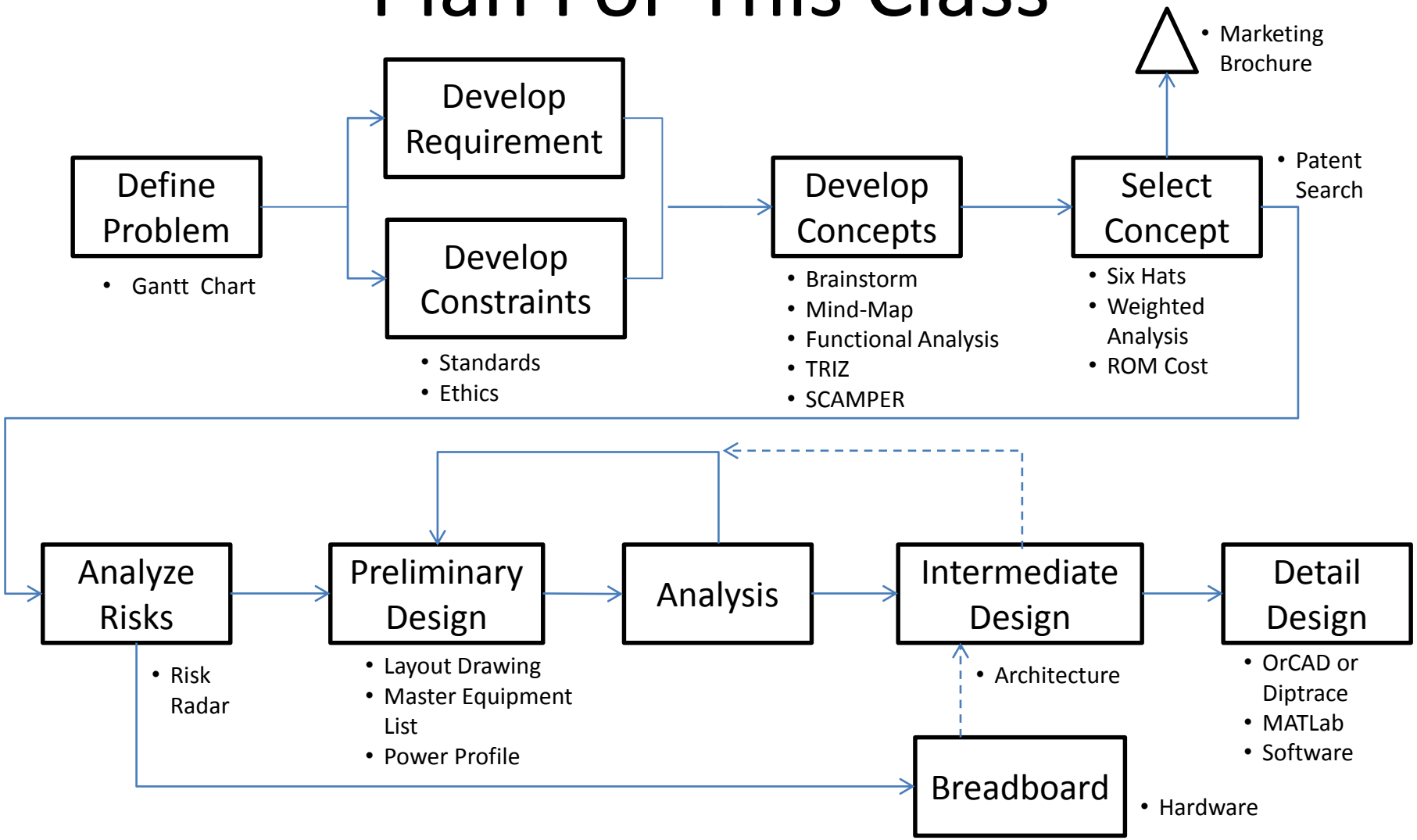
Elec 4309 Senior Design

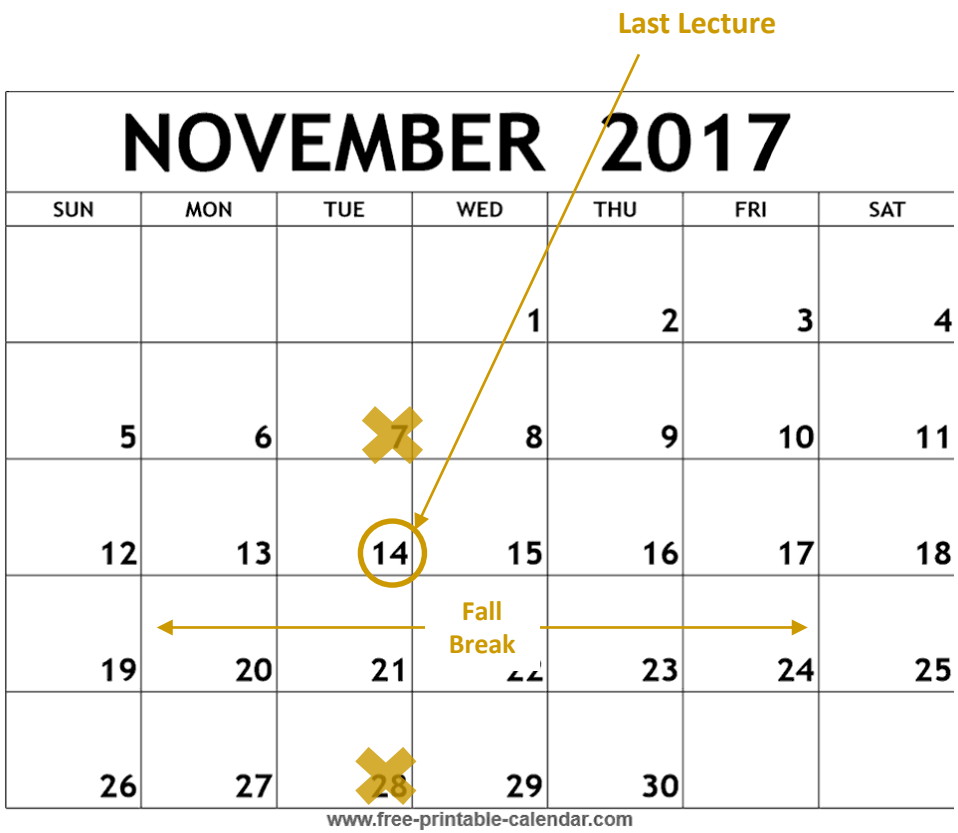
Wendell H Chun

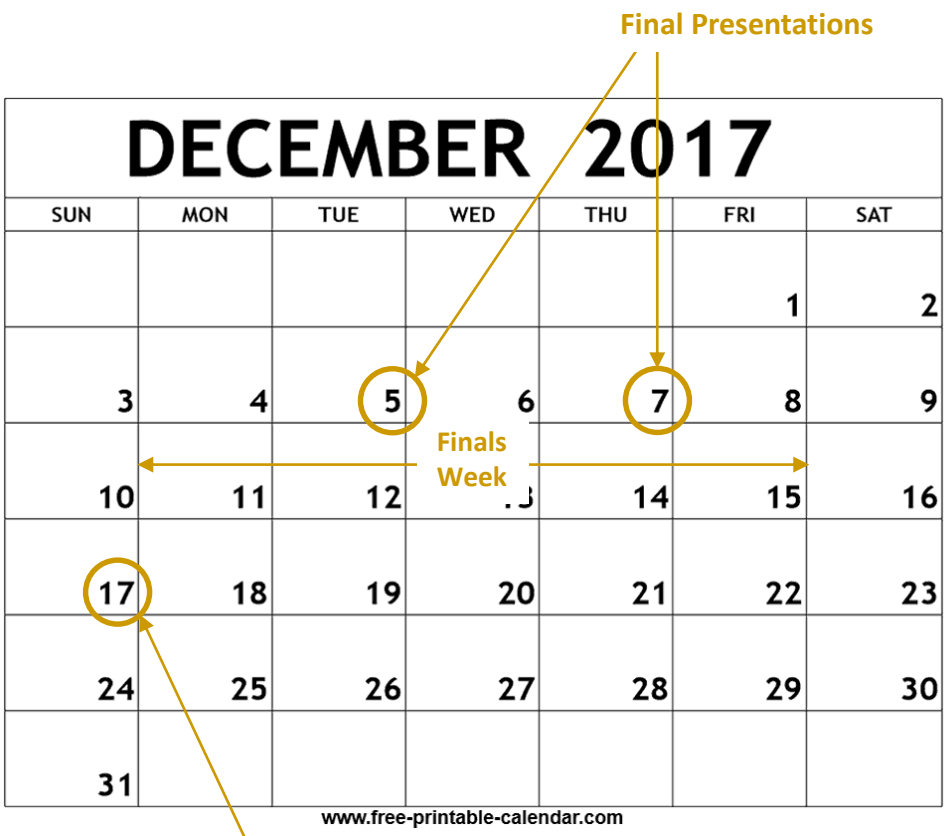
Oct. 31, 2017

Plan For This Class



Remaining Class Schedule





- No Class (Out of Town)

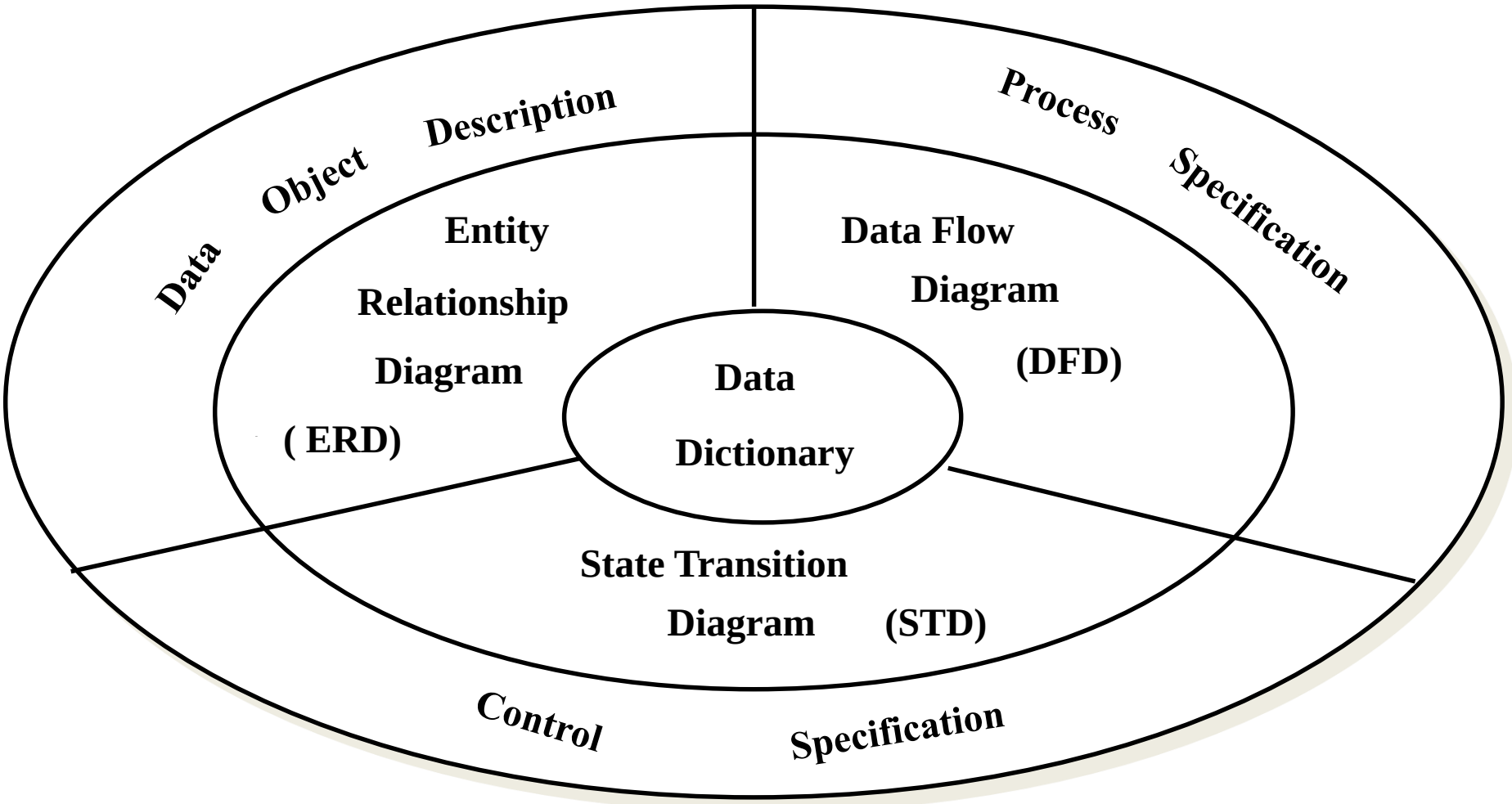
Analysis Modeling

- Two primary methods today:
  - Structured Analysis
  - Object-oriented analysis
- Some important considerations:
  - Analysis products must be maintainable
  - Effective partitioning is essential
  - Graphics should be used whenever possible
  - Distinguish between logical and implementation

Structured Analysis

- Elements of Analysis:
  - Describe what customer requires
  - Establish basis for creating (software) design
  - Define requirements ***that can be validated***

Graphical View of Model

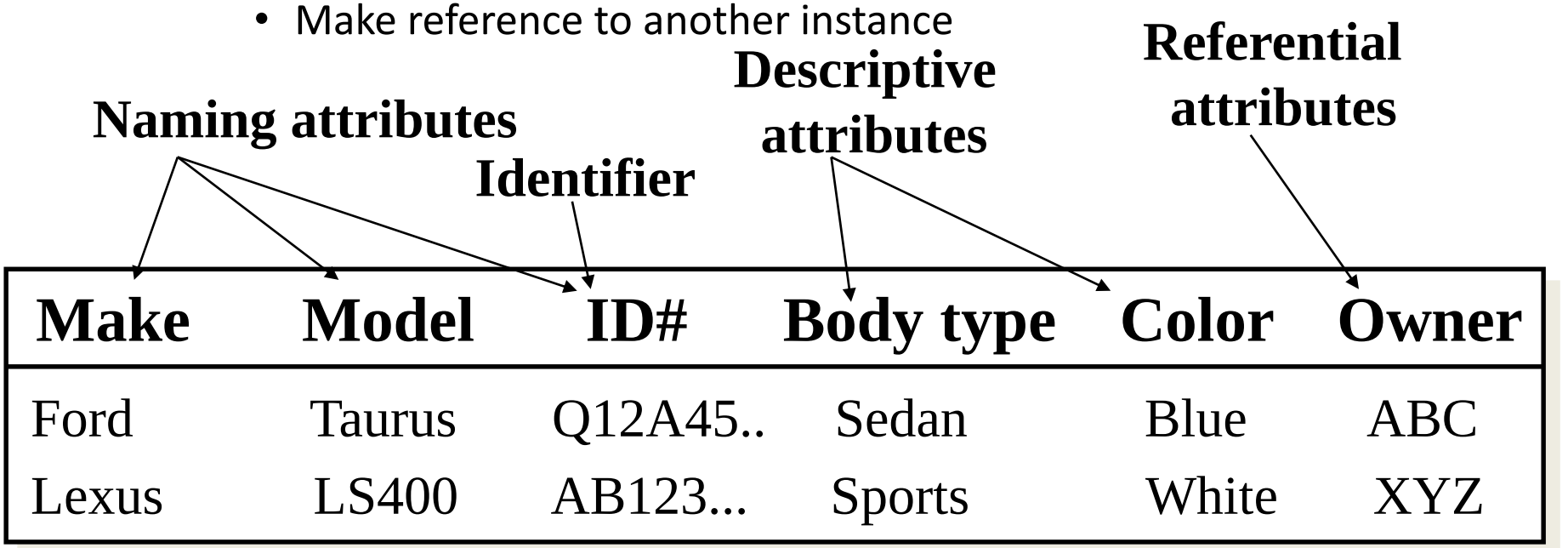


Data Modeling

- The model consists of:
  - Data object [types]
  - Attributes
  - Relationships
- Data objects:
  - A representation of almost any composite information that must be understood by software.

Data Modeling

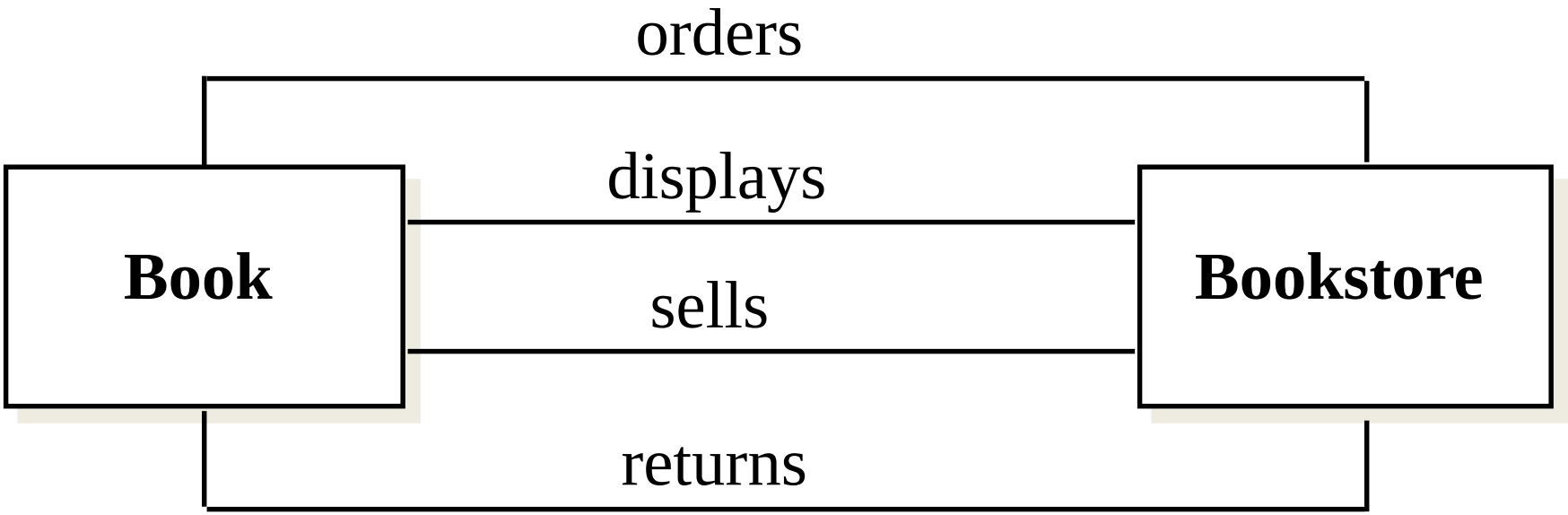
- Attributes
  - Attributes define the properties of a data object and take on one of three different characteristics:
    - Name an instance of the data object
    - Describe the instance
    - Make reference to another instance



Make	Model	ID#	Body type	Color	Owner
Ford	Taurus	Q12A45..	Sedan	Blue	ABC
Lexus	LS400	AB123...	Sports	White	XYZ

Data Modeling

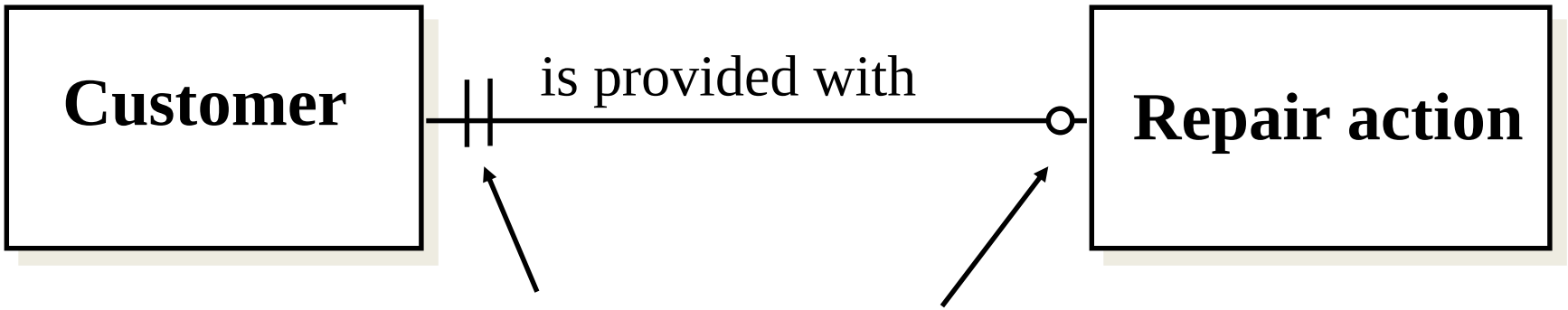
- Relationships
  - Defined pairwise -- many varieties



Cardinality and Modality

- Cardinality
  - How many occurrences of object X are related to how many occurrences of object Y
    - One-to-one (1:1)
    - One-to-many (1:N)
    - Many-to-many (M:N)
- Modality
  - = 0 → optional relationship
  - = 1 → relationship must appear

Example

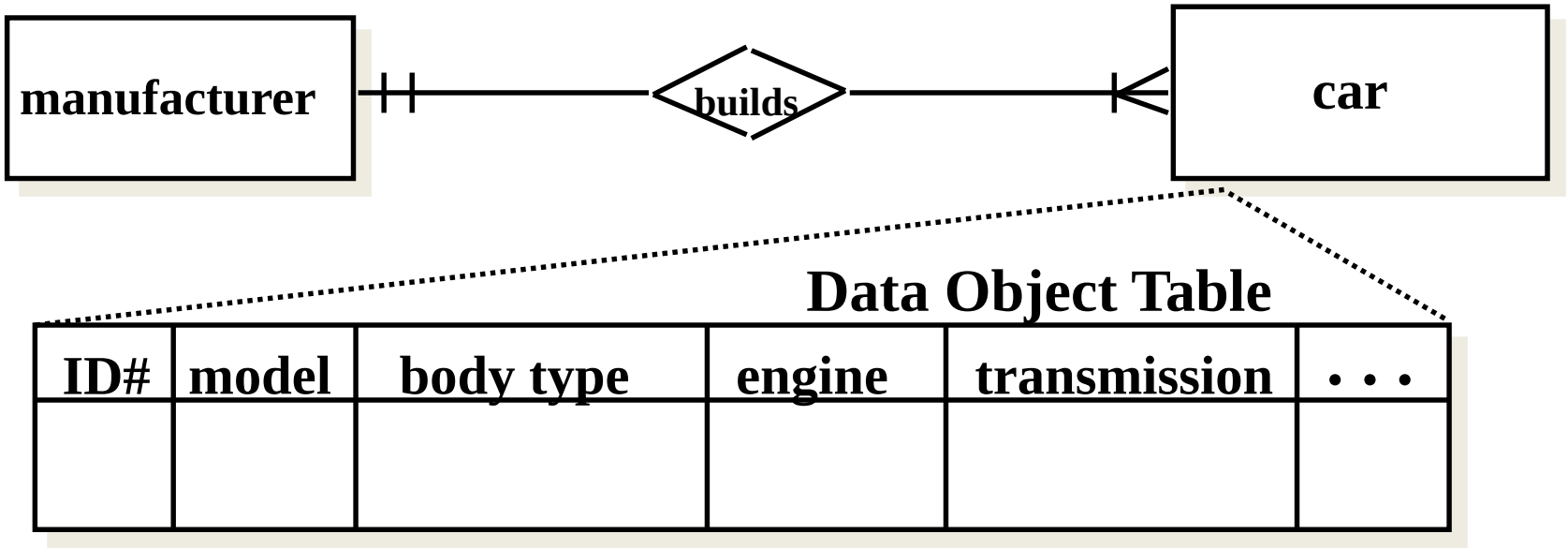


Mandatory: in order to have a repair action, we must have a customer

Optional: there may be a situation in which a repair action is not necessary

Entity Relation Diagrams (ERD)

- Cornerstone of the data model – includes:
  - data objects,
  - attributes,
  - relationships, and
  - various type indicators



ERD Example

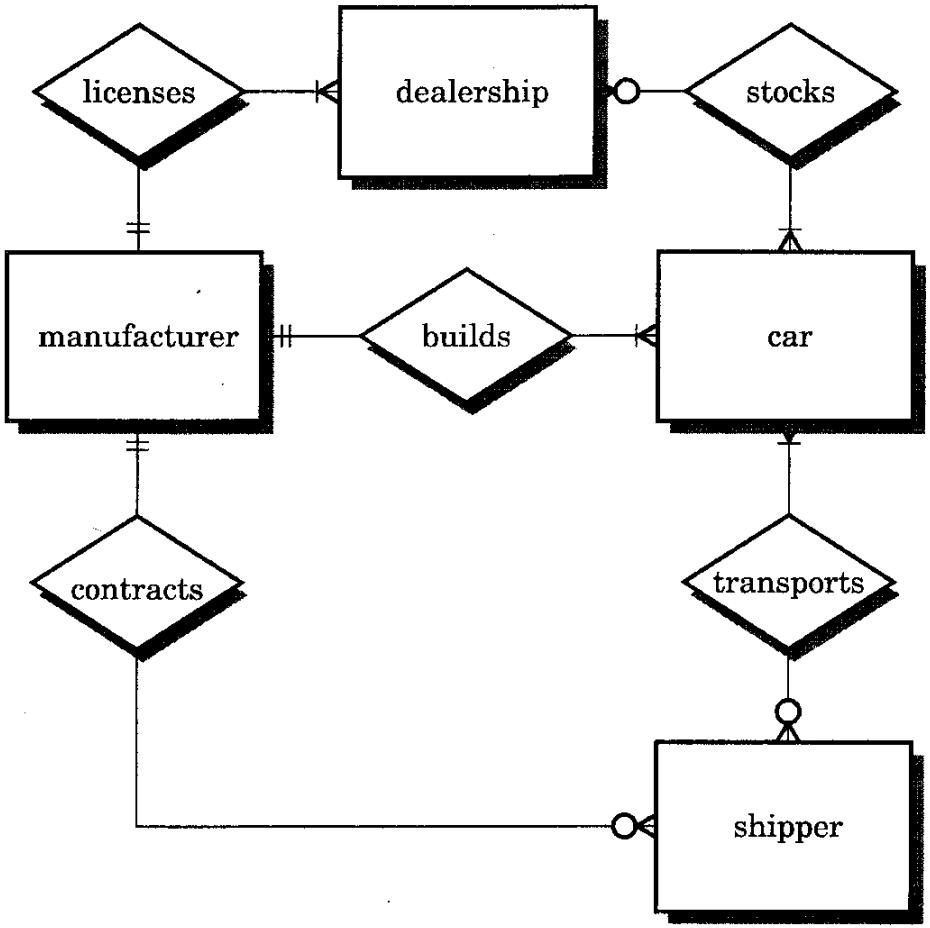


FIGURE
An expanded ERD

Data Object Hierarchies

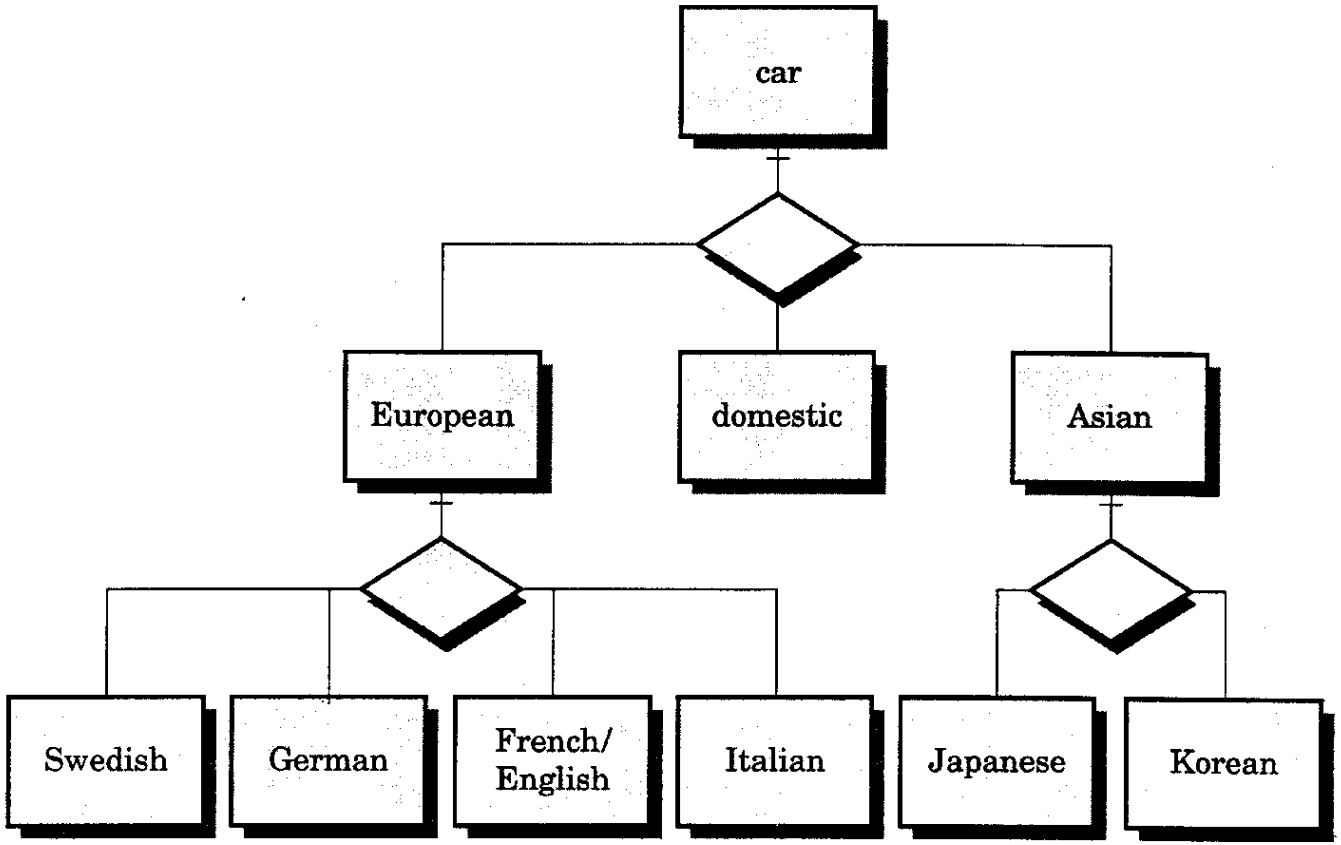


FIGURE Data object type hierachies

Associating Data Objects

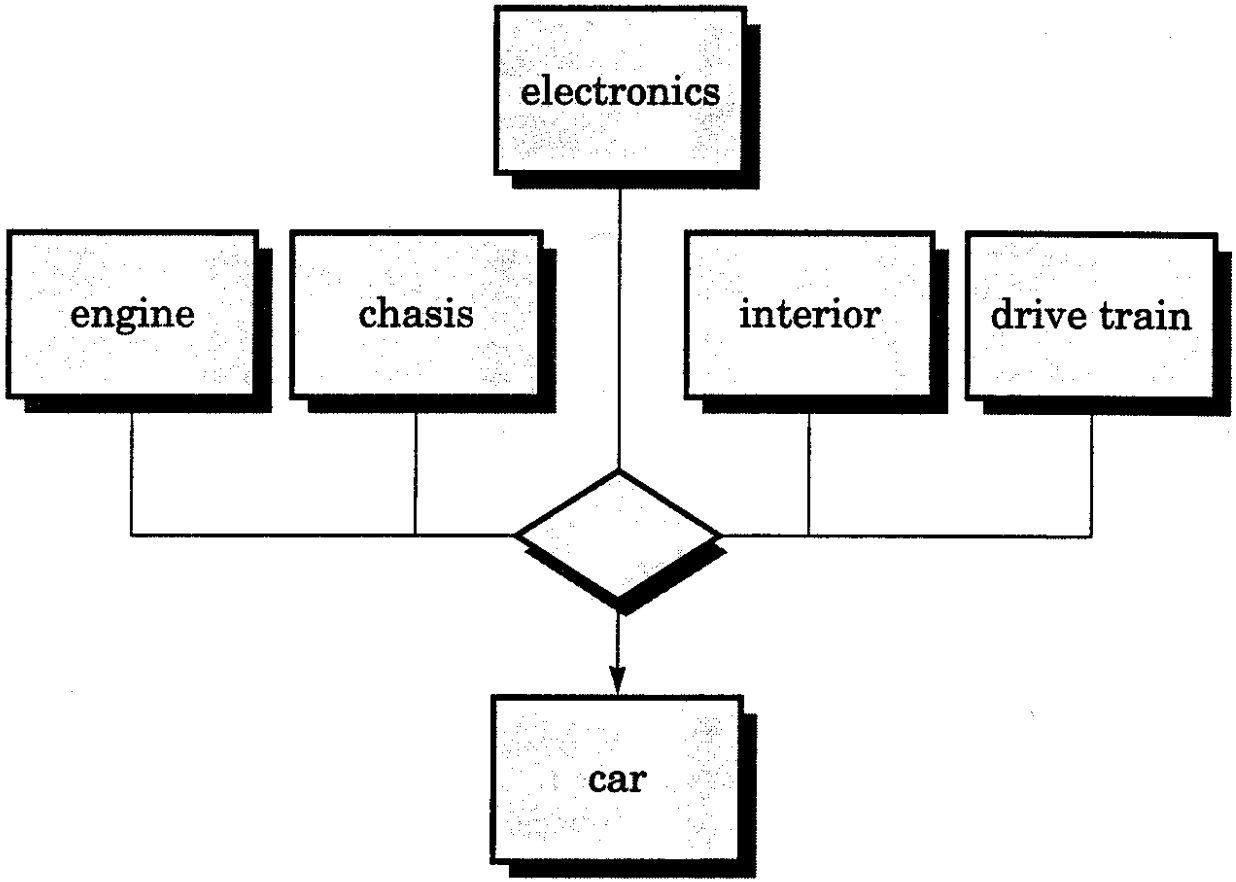
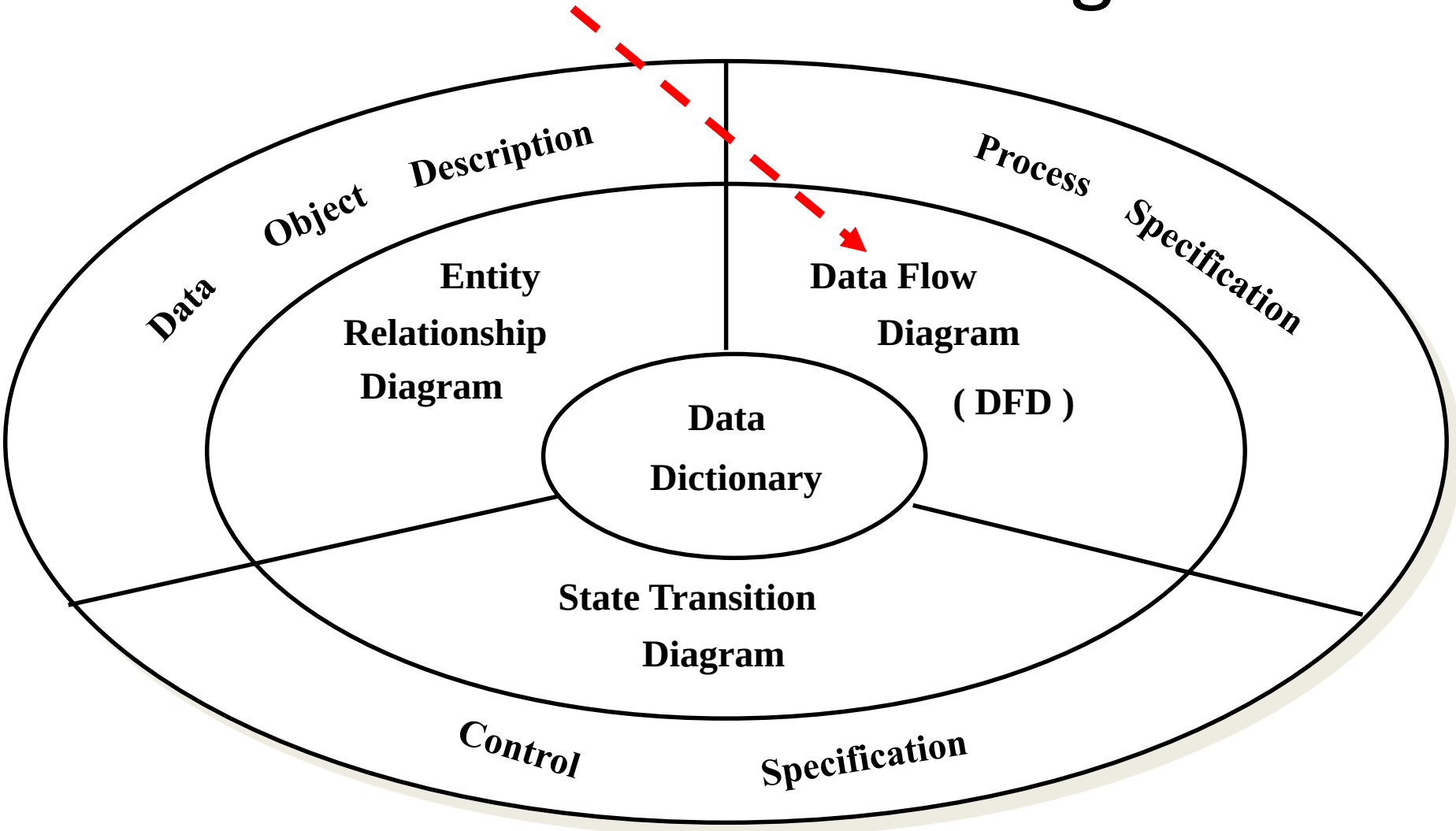


FIGURE
Associating data objects

Functional Modeling



Data Flow Diagrams (DFD)

- A graphical technique that depicts information flow and the transforms applied as data move from input to output
- ***Not*** the same as flow charts. Does not show the logic of the transformations
- Can be used at any level of abstraction

General Information Flow Model

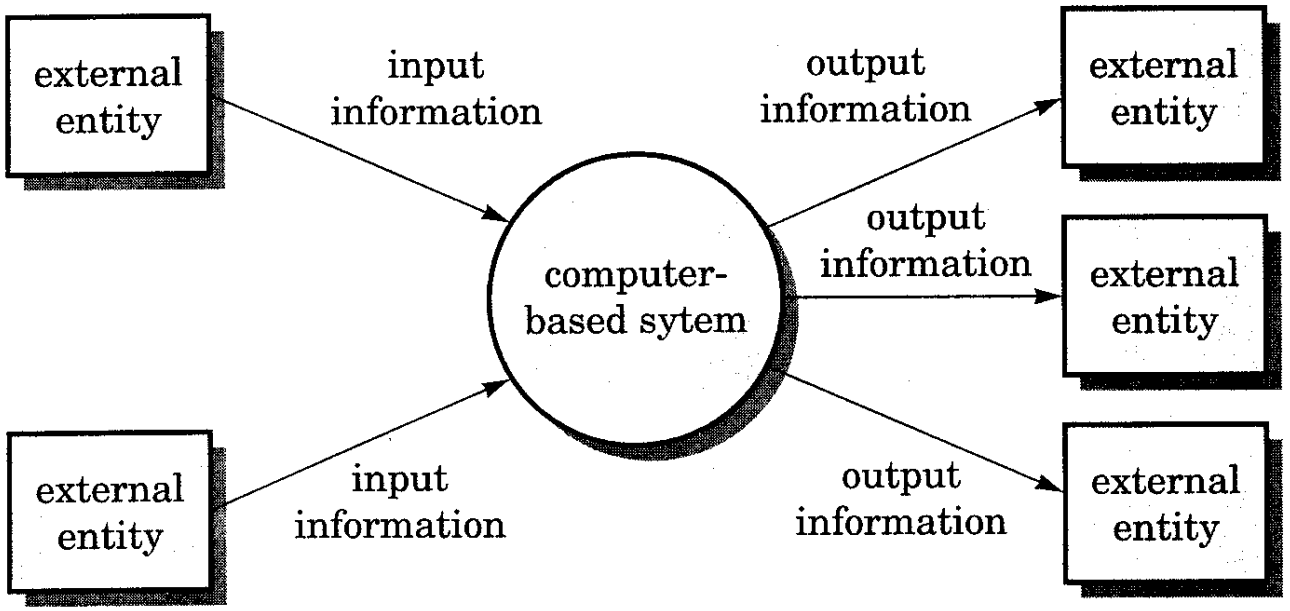
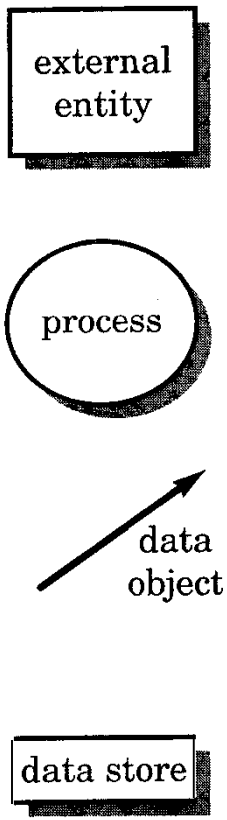


FIGURE
Information flow model

Basic Notation



A producer or consumer of information that resides outside the bounds of the system to be modeled

A transformer of information (a function) that resides within the bounds of the system to be modeled

A data object; the arrowhead indicates the direction of data flow

A repository of data that is to be stored for use by one or more processes; may be as simple as a buffer or queue or as sophisticated as a relational database

FIGURE
Basic DFD notation

External Entity

A producer or consumer of data

Examples: a person, a device, a sensor

Another example: computer-based system

Data must always originate somewhere and must always be sent to something

Process

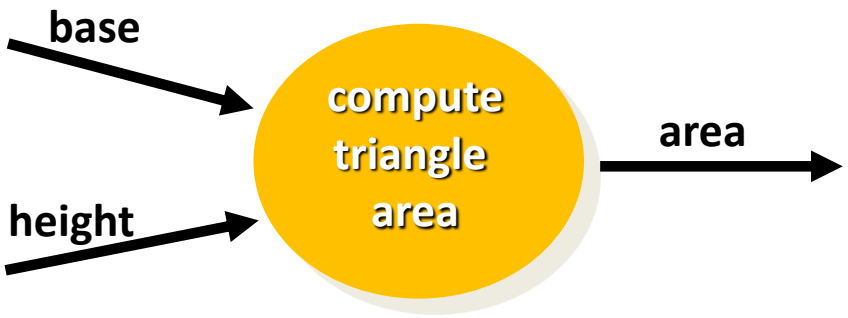
A data transformer (changes input to output)

Examples: compute taxes, determine area, format report, display graph

Data must always be processed in some way to achieve system function

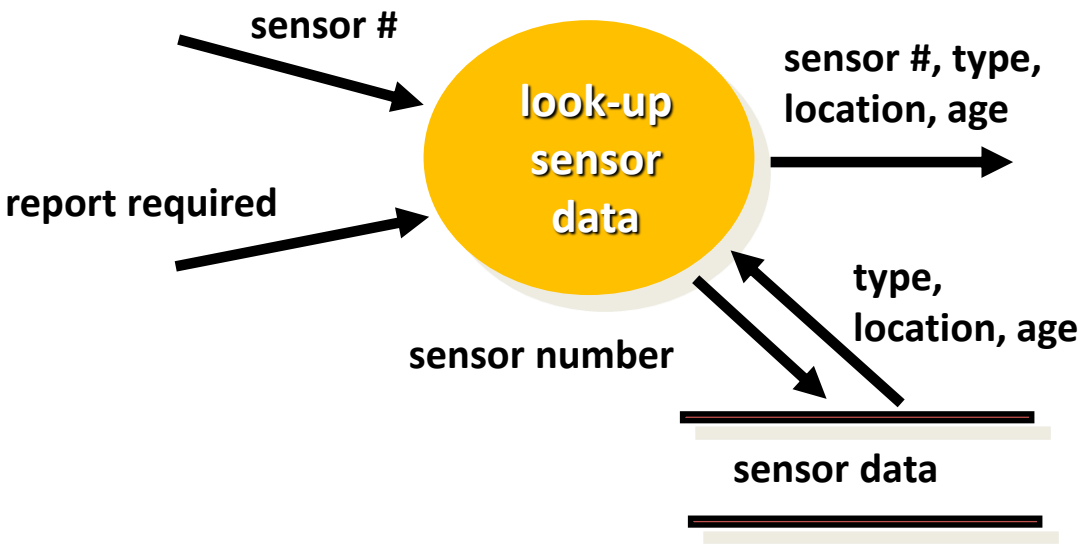
Data Flow

Data flows through a system, beginning as input and be transformed into output.



Data Stores

Data is often stored for later use.



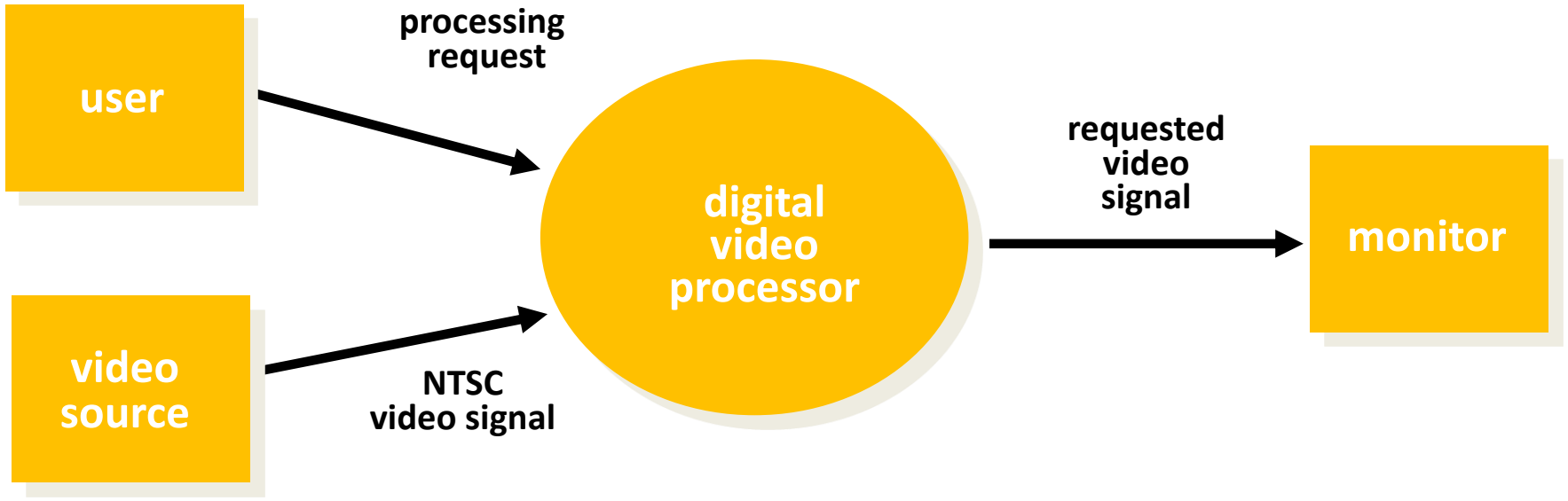
Data Flow Diagramming: Guidelines

- All icons must be labeled with meaningful names
- The DFD evolves through a number of levels of detail
- Always begin with a context level diagram (also called level 0)
- Always show external entities at level 0
- Always label data flow arrows
- Do not represent procedural logic

Constructing a DFD - I

- Review the data model to isolate data objects and use a grammatical parse to determine “operations”
- Determine external entities (producers and consumers of data)
- Create a level 0 DFD

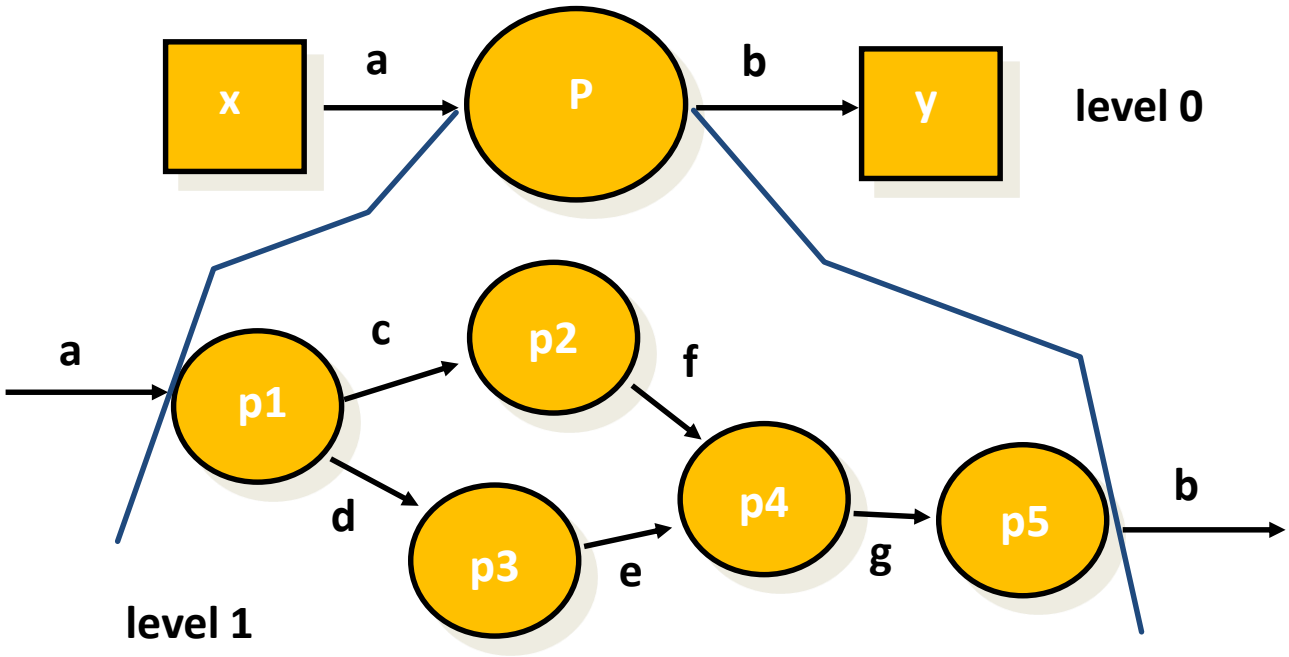
Level 0 DFD Example



Constructing a DFD - II

- Write a narrative describing the transform
- Parse to determine next level transforms
- “balance” the flow to maintain data flow continuity
- Develop a level 1 DFD
- Use a 1:5 (approx.) expansion ratio

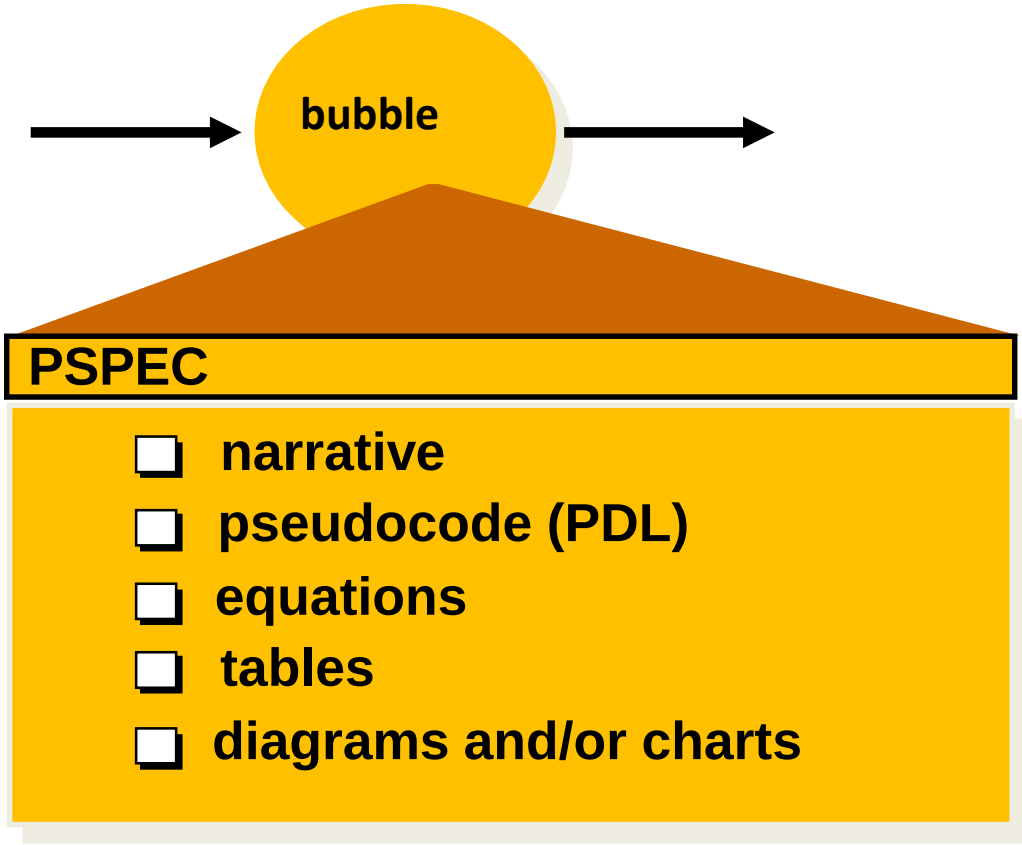
The Data Flow Hierarchy



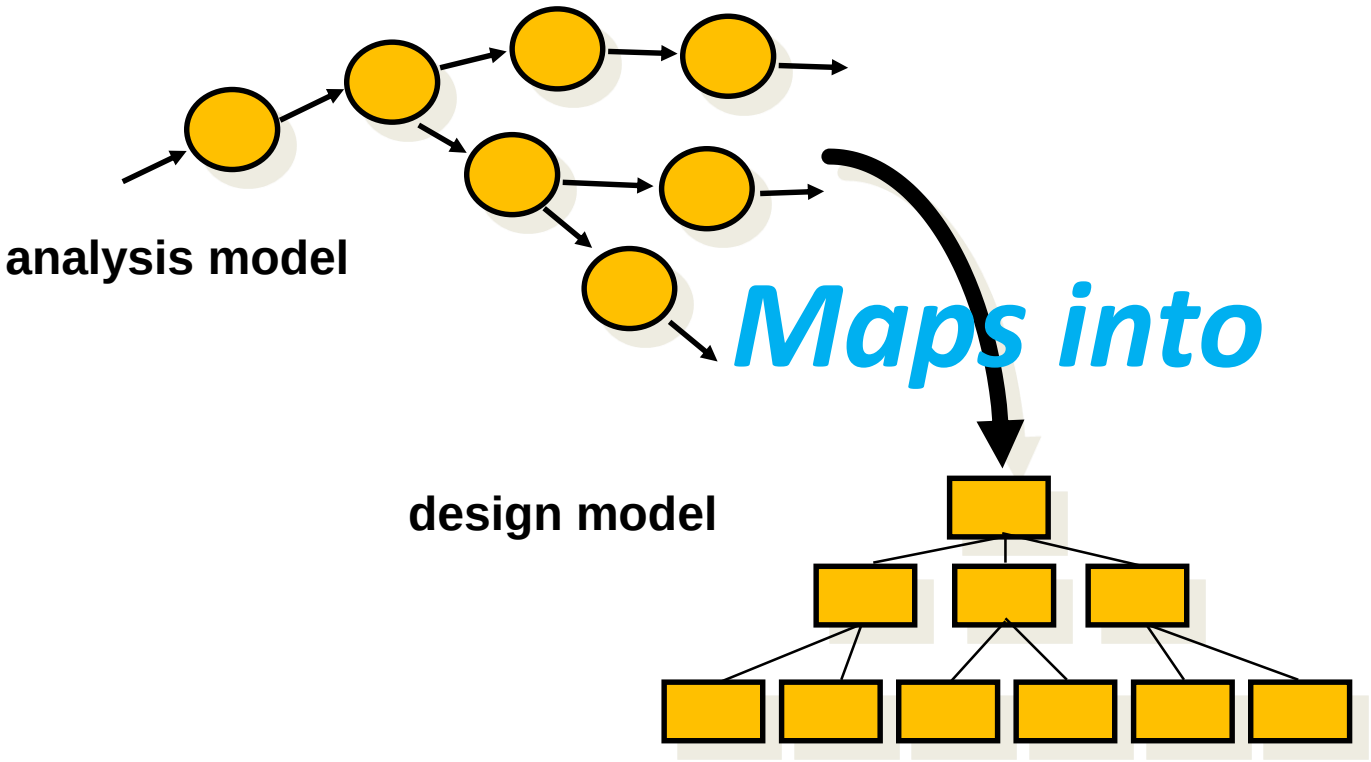
Flow Modeling Notes

- Each bubble is refined until it does just one thing
- The expansion ratio decreases as the number of levels increase
- Most systems require between 3 and 7 levels for an adequate flow model
- A single data flow item (arrow) may be expanded as levels increase (data dictionary provides information)

Process Specification (PSPEC)

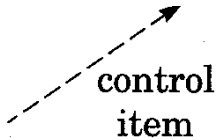


DFDs: A Look Ahead



Real Time Extensions

- Fundamental issue - The time at which results are produced is a part of the correctness of the computation.
- Hatley/Pirbhai notation:

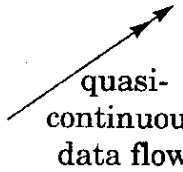


A control item or event; takes on a boolean or discrete value; the arrowhead indicates the direction of data flow.

FIGURE
Extended structured analysis notation for real-time systems developed by Hatley and Pirbhai [HAT87]

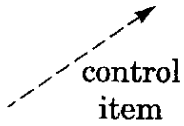
The vertical bar is a reference to a control specification (CSPEC) that describes the behavior of a system and defines how processes are activated as a consequence of events.

Ward/Mellor Notation

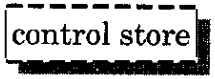


A data object that is input or output from a process on a "continuous" basis

A transformer of control or "events"; accepts control and input and produces control as output



A control item or event; takes on a boolean or discrete value; the arrowhead indicates the direction of data flow



A repository of control items that are to be stored for use by one or more processes

Multiple equivalent instances of the same process; used when multiple processes are created in multitasking system

FIGURE
Extended structured analysis notation for real-time systems developed by Ward and Mellor [WAR85]

Example

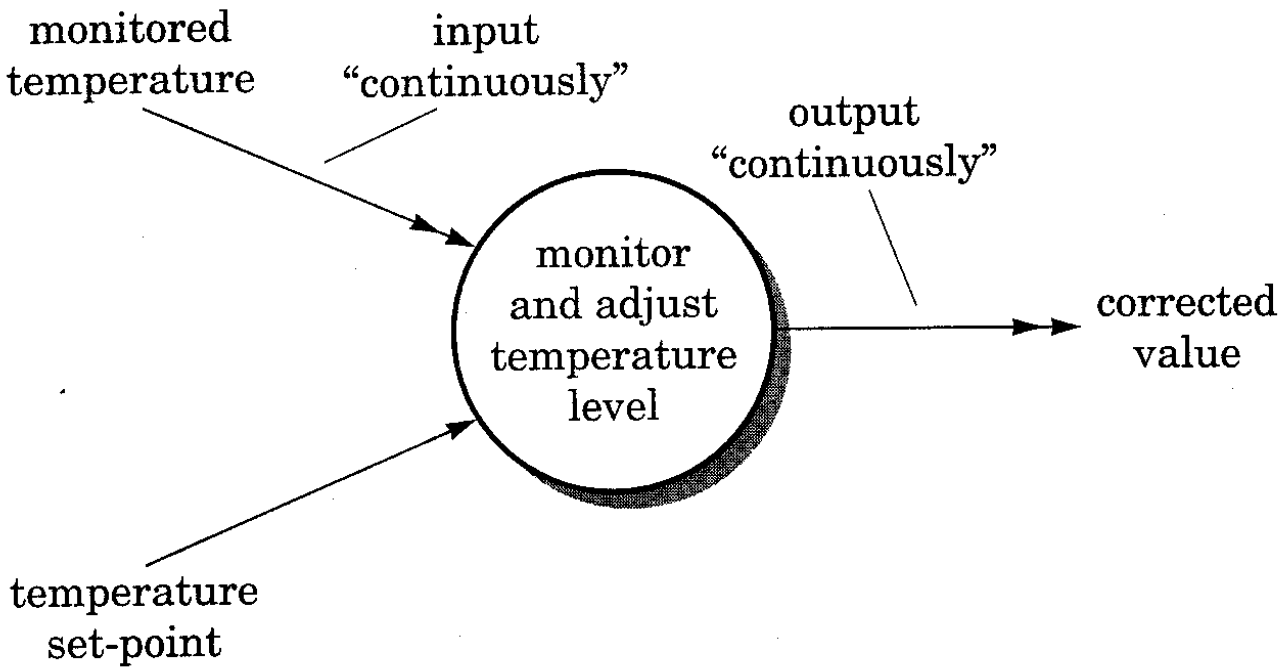


FIGURE
Time-continuous data flow

Example

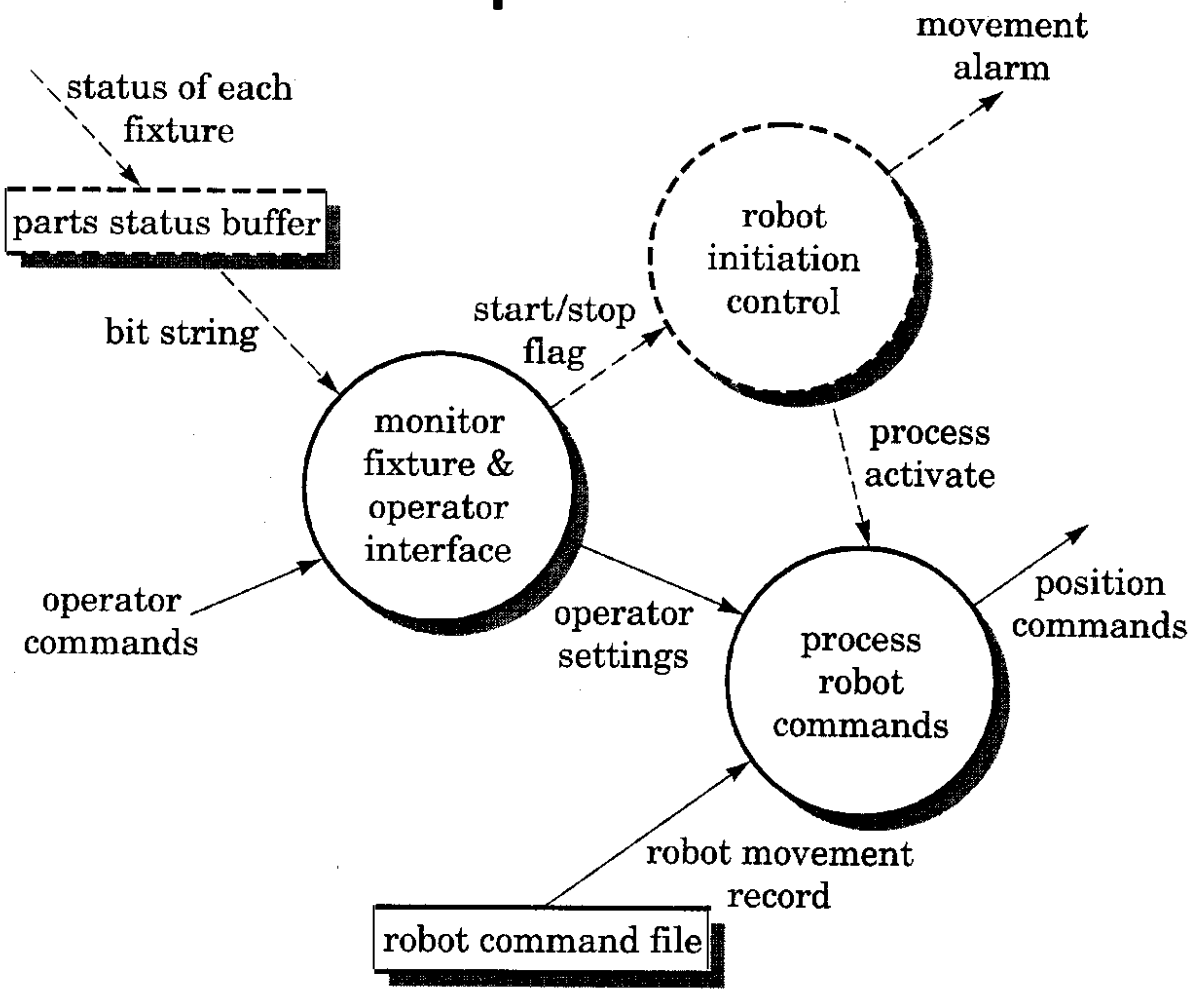


FIGURE
Data and control flows using Ward and Mellor [WAR85] notation

Hatley and Pirbhai Extensions

- Use separate *data flow diagram* (DFD) and *control flow diagram* (CFD)
- Data flow diagrams:
  - Used to represent data and the processes that manipulate it
- Control flow diagrams:
  - Show how events flow among processes and show those external events that cause various processes to be activated

Control Flow Diagrams

- Represents "events" and the processes that manage events
- An "event" is a Boolean condition that can be ascertained by:
  - Listing all sensors that are "read" by the software.
  - Listing all interrupt conditions.
  - Listing all "switches" that are actuated by an operator.
  - Listing all data conditions.
  - Recalling the noun/verb parse that was applied to the processing narrative, review all "control items" as possible CSPEC inputs/outputs.

The Control Model

- The control flow diagram is "superimposed" on the DFD and shows events that control the processes noted in the DFD
- Control flows—events and control items—are noted by dashed arrows
- A vertical bar implies an input to or output from a control spec (CSPEC) — a separate specification that describes how control is handled
- A dashed arrow entering a vertical bar is an input to the CSPEC
- A dashed arrow leaving a process implies a data condition
- A dashed arrow entering a process implies a control input read directly by the process

Relationship Between Models

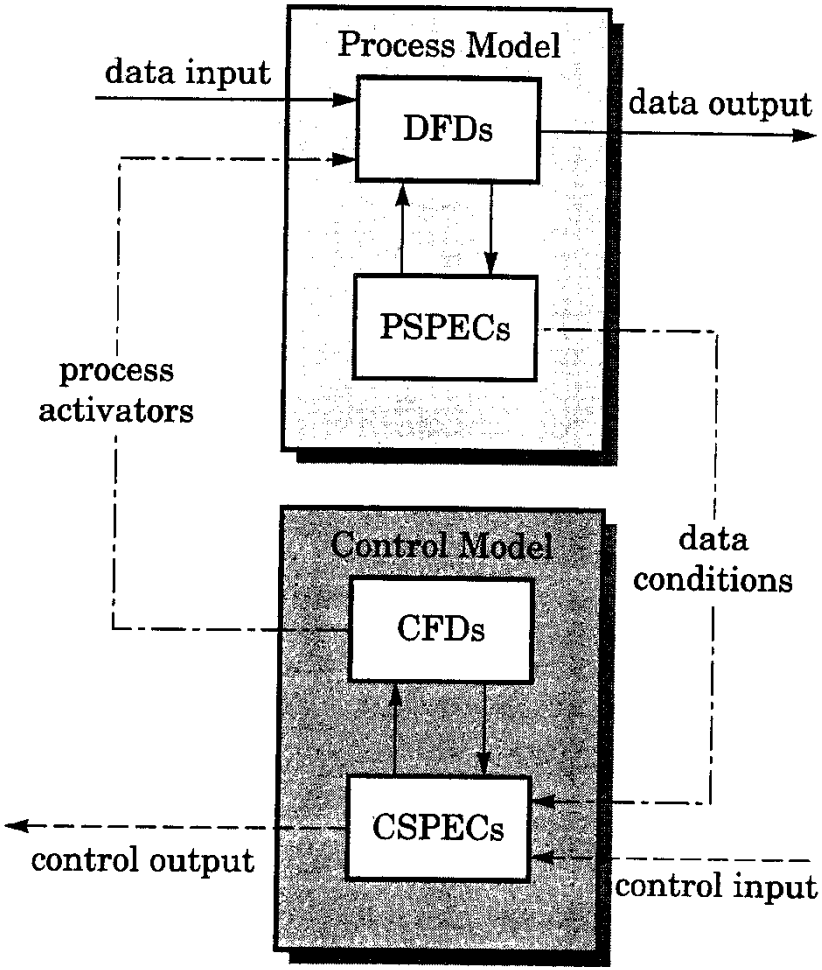


FIGURE
The relationship between data and control models [HAT87]

Example

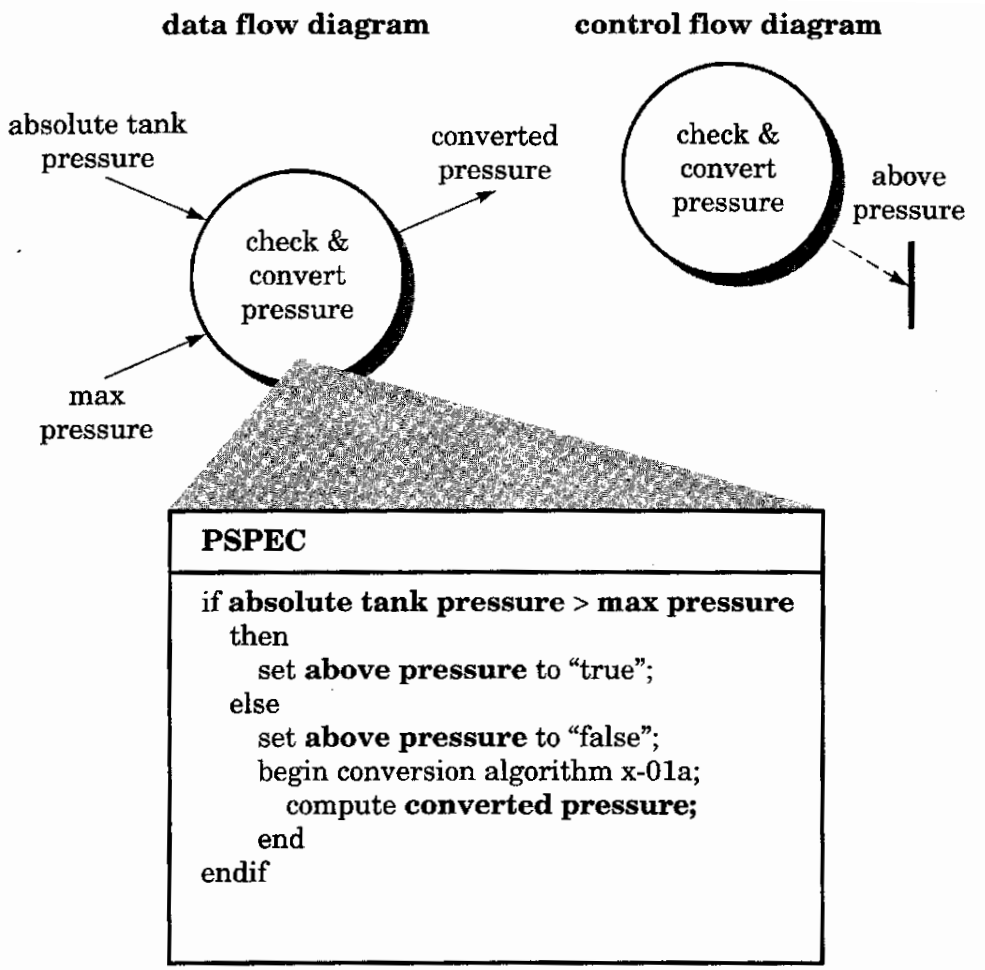
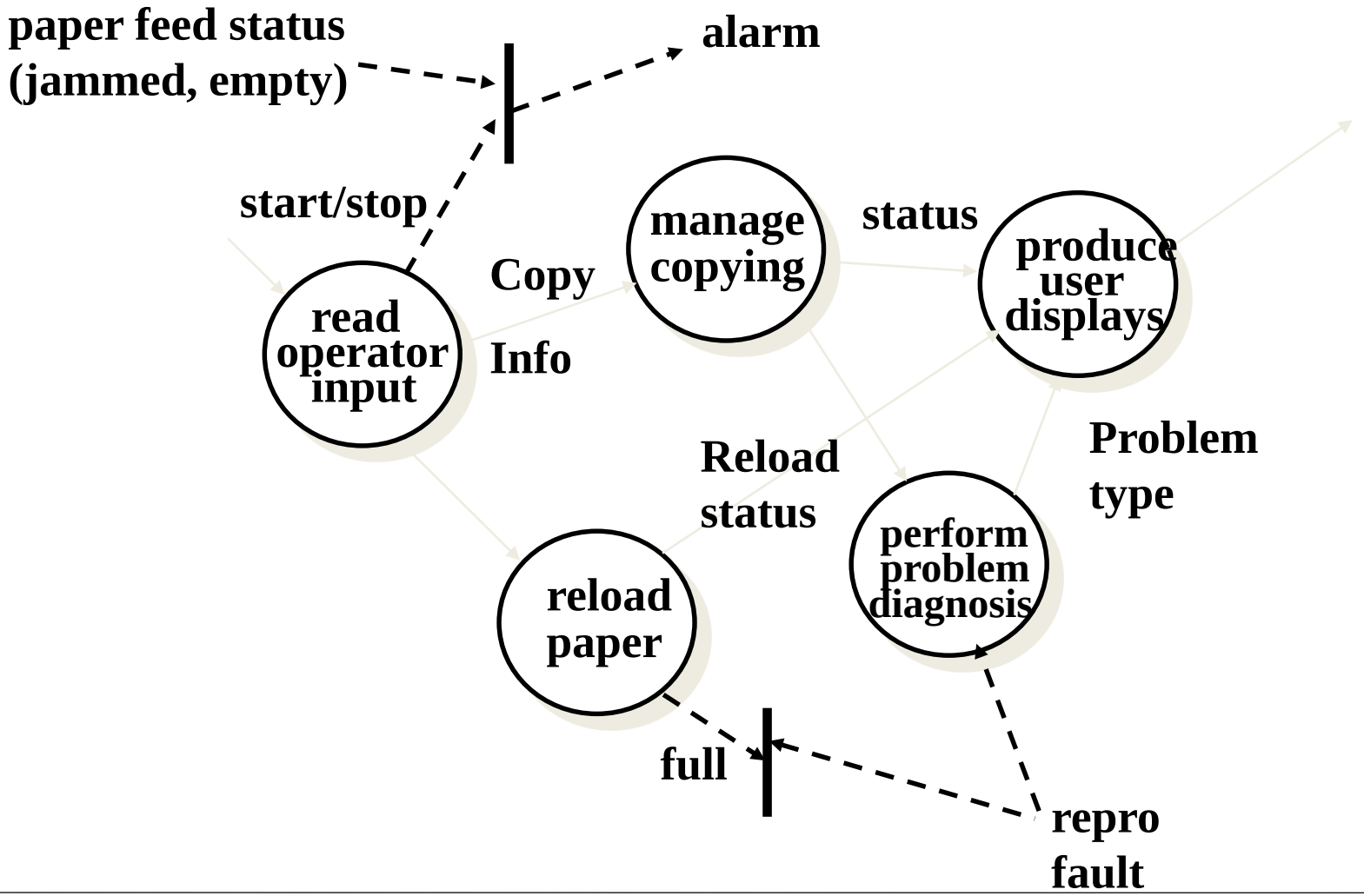


FIGURE
Data conditions

CFD for Photocopier



Control Specification (CSPEC)

The CSPEC can be:

- ❑ state diagram (sequential spec)

State Transition Diagram (STD)

- ❑ state transition table

State Event Matrices (SEM)

- ❑ decision tables (DT)

- ❑ activation tables

Process Activation Tables (PAT)

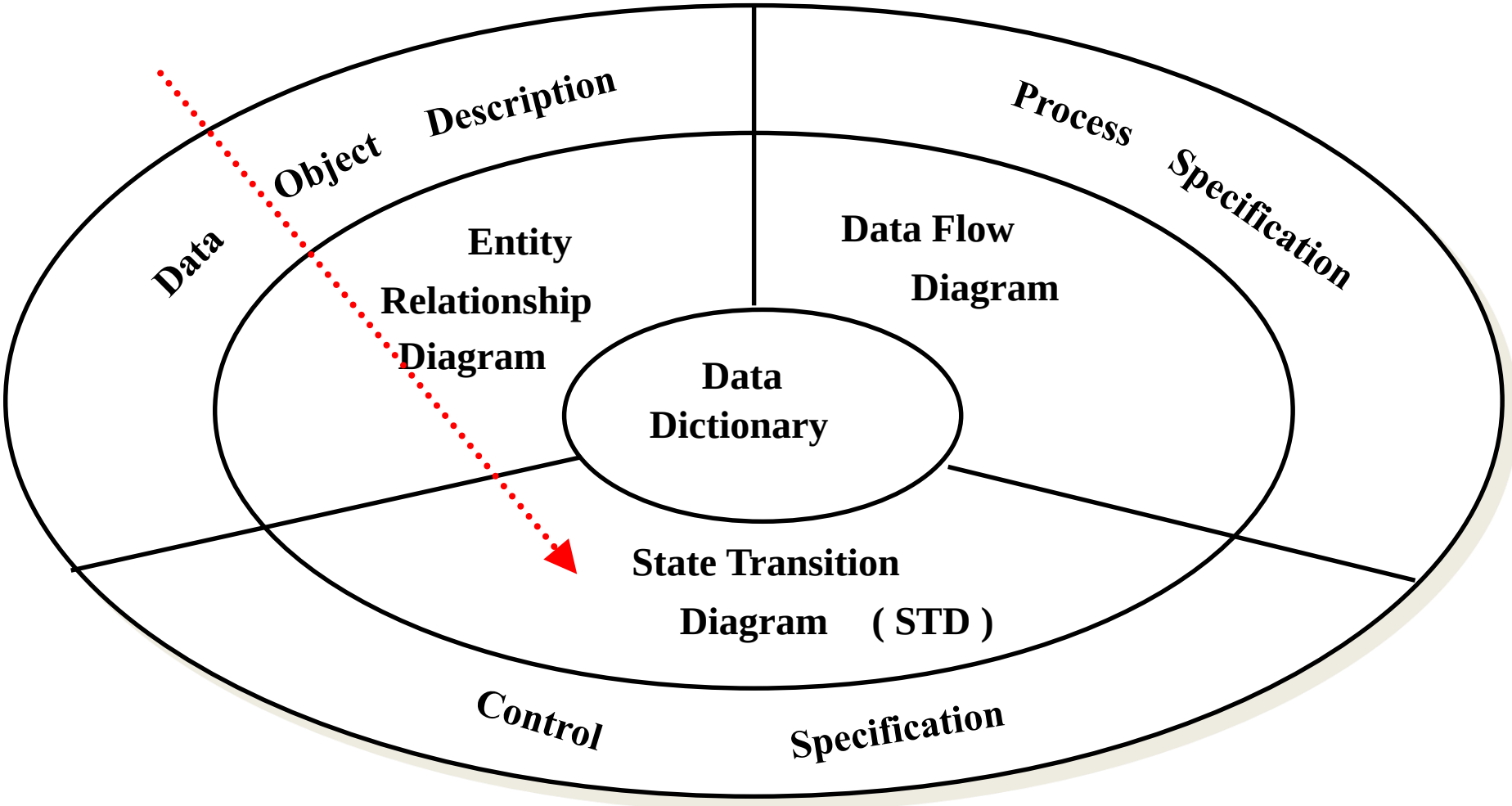
combinatorial spec

- **Finite State Machines (FSMs) can be used only when a finite number of inputs having a finite set of values can lead to a finite number of outputs (or set of actions) having a finite set of values.**

Guidelines for Building a CSPEC

- list all sensors that are "read" by the software
- list all interrupt conditions
- list all "switches" that are actuated by the operator
- list all data conditions
- recalling the noun-verb parse that was applied to the software statement of scope, review all "control items" as possible CSPEC inputs/outputs
- describe the behavior of a system by identifying its states; identify how each state is reach and defines the transitions between states
- focus on possible omissions ... a very common error in specifying control, e.g., ask: "Is there any other way I can get to this state or exit from it?"

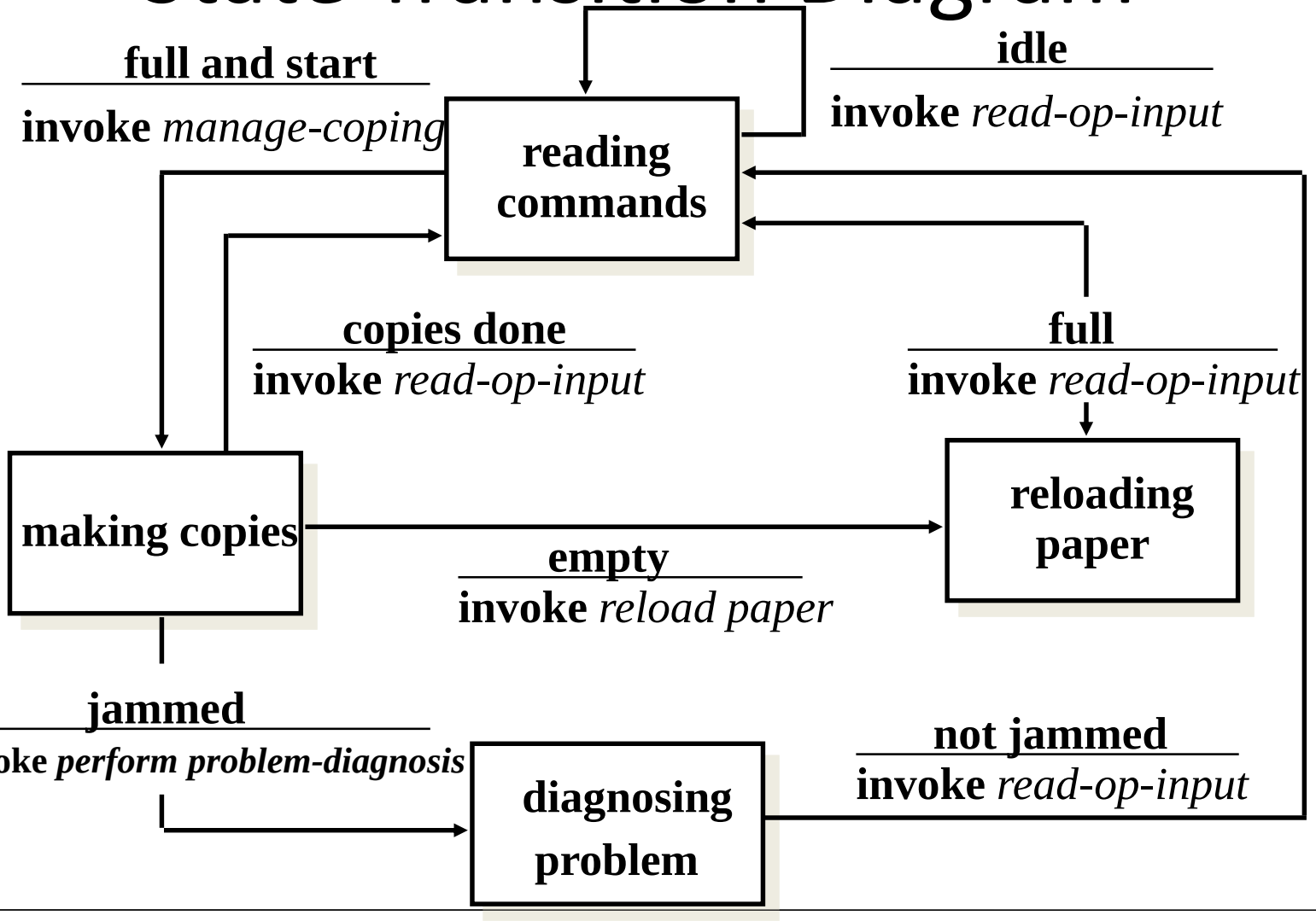
Behavioral Modeling



State Transition Diagrams

- A State is any observable mode of behavior
  - e.g., reading commands, computing control, waiting for next time event
- States represented as rectangles
- Arrows represent transitions
- Value above arrow identifies event causing transition
- Value below arrow indicates ensuring action

State Transition Diagram



Creating an ERD

- List entities that customer addresses
- For each, determine the connections
- For each connection, create one or more object-relationship pairs
- For each relationship, determine cardinality and modality
- Define the attributes of each entity
- Formalize and review ERD
- Iterate

Home Security System Example

- Initial entities
  - Homeowner, control panel, sensors, security system and monitoring service

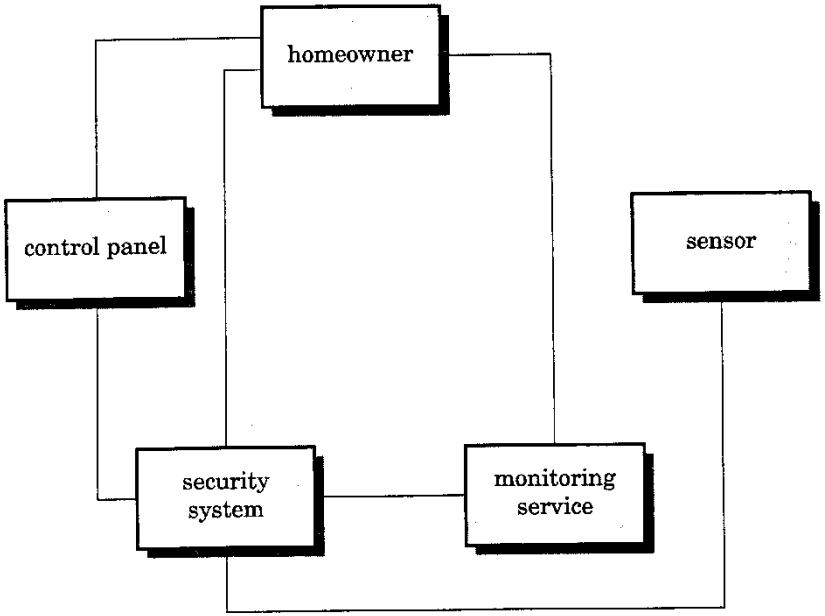


FIGURE
Establishing connections

Home Security System Example

- Relationships between sensor and security systems:
  - Security system monitors sensor
  - Security system enables/disables sensor
  - Security system tests sensor
  - Security system programs sensor

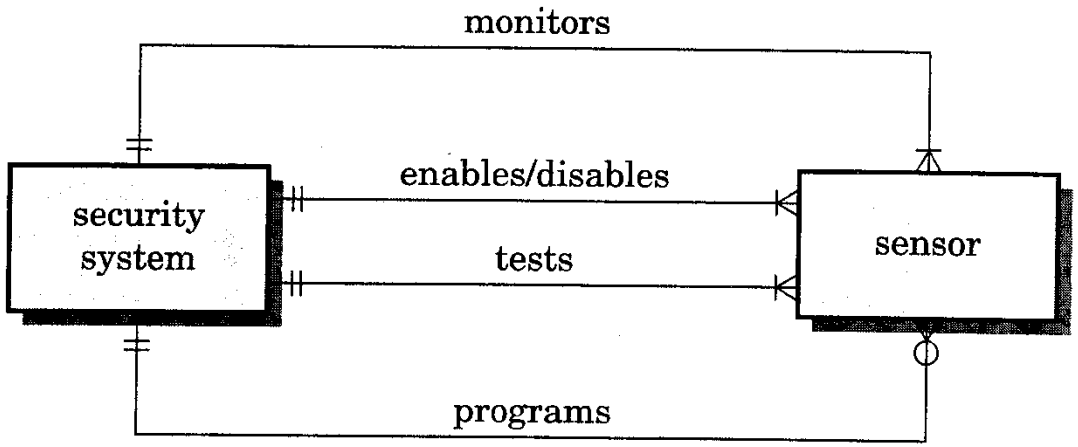


FIGURE
Developing relationships and cardinality/modality

Creating a Data Flow Model

- First create level 0 diagram:
  - Depict software system as single bubble
  - Show primary inputs and outputs
- Identify processes, data objects, and data stores to be expanded at next level
- Label all arrows with meaningful names
- Information flow continuity must be maintained
- Refine only one bubble at a time

Home Security System Example

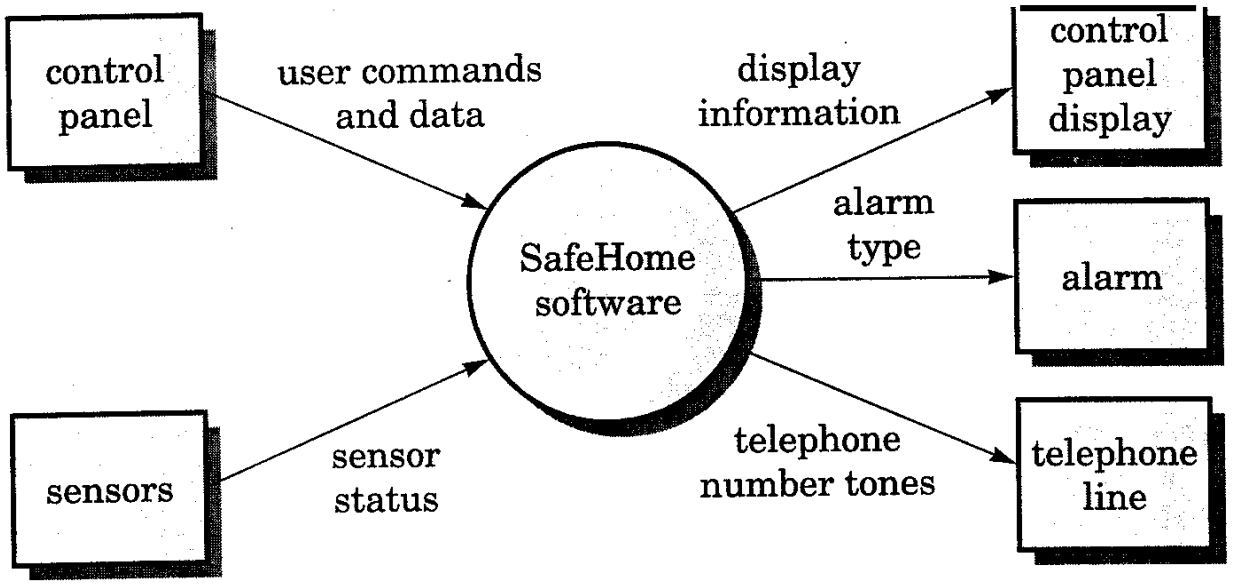


FIGURE
Context-level DFD for SafeHome

Refinement

- Analyze textual description of bubble:
  - Verbs are often processes
  - Nouns are often external entities, data or control objects or data stores
- Examples:
  - Control panel is used to program and configure the system
  - Upon a sensor event, the software invokes an alarm

Home Security System Example

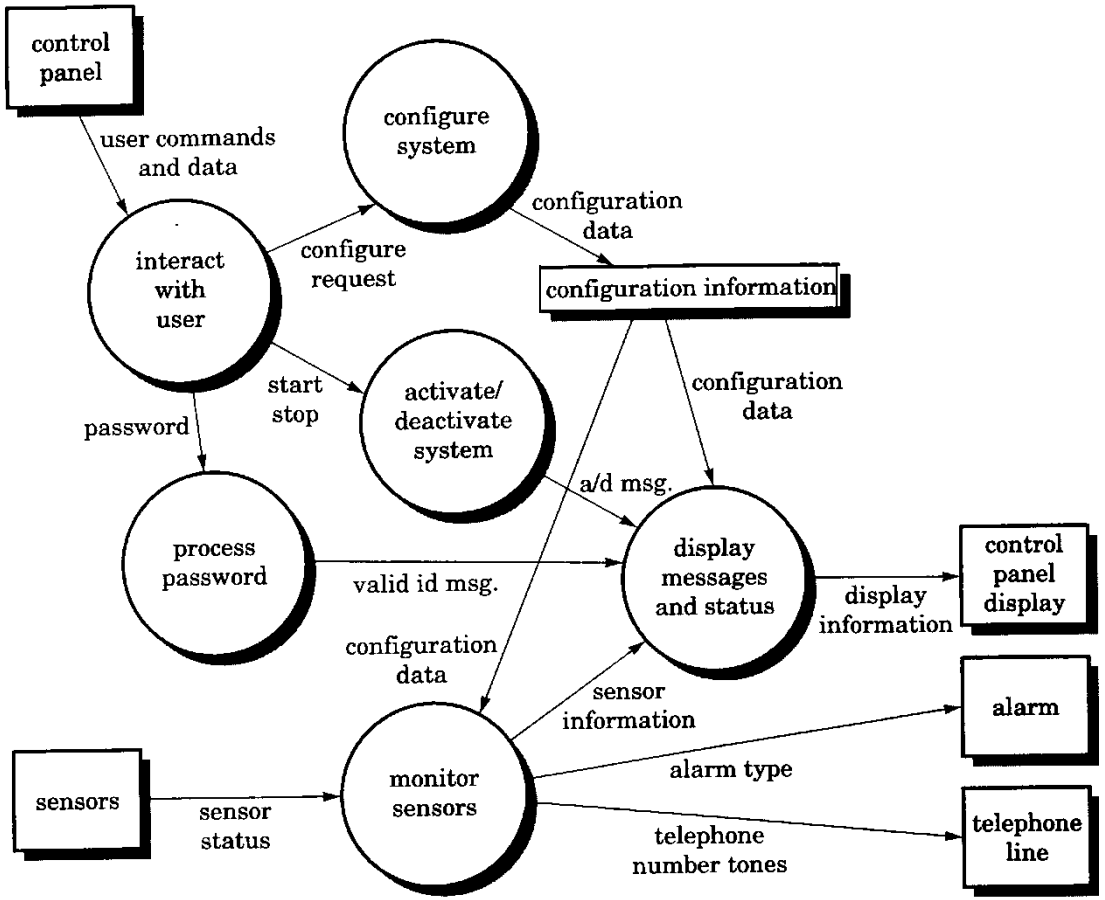


FIGURE Level 1 DFD for SafeHome

Home Security System Example

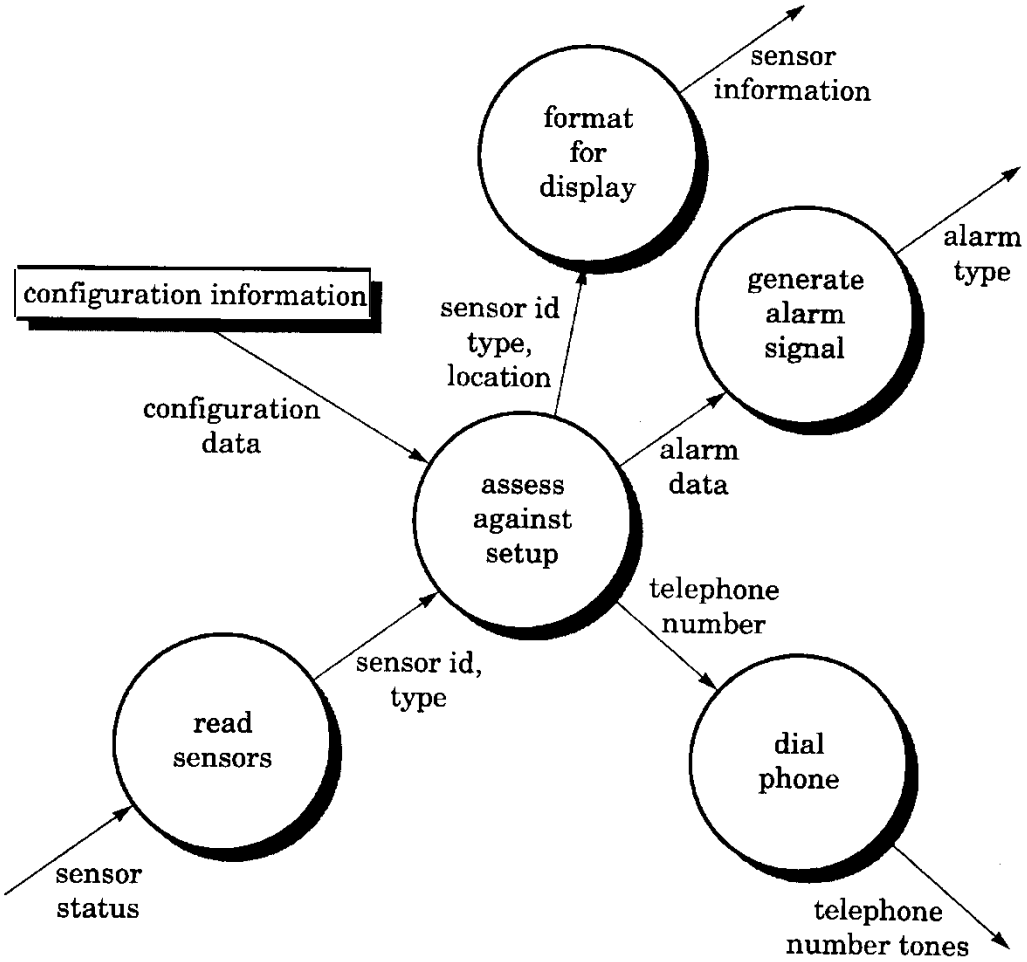


FIGURE
Level 2 DFD that refines the *monitor sensors* process

Creating Control Flow Models

- Strip arrows from DFD
- Add event and control items. E.g., try:
  - List all sensors read by the software
  - List all interrupt conditions
  - List all operator actuated switches
  - List all data conditions
  - Check noun-verb parse for possible CSPEC I/O
  - Identify states, how each is reached and transitions
  - Focus on possible omissions

Level 1 CFD for Safe-Home

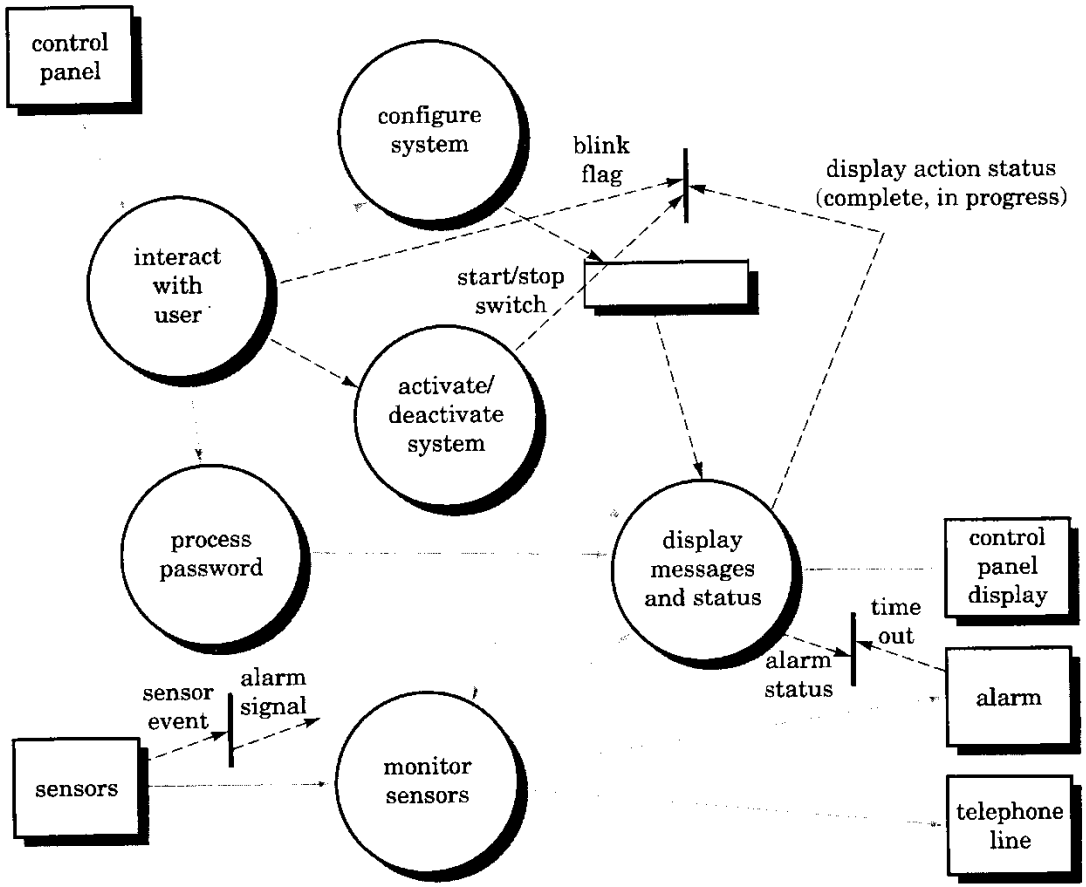


FIGURE Level 1 CFD for SafeHome

Control Specification

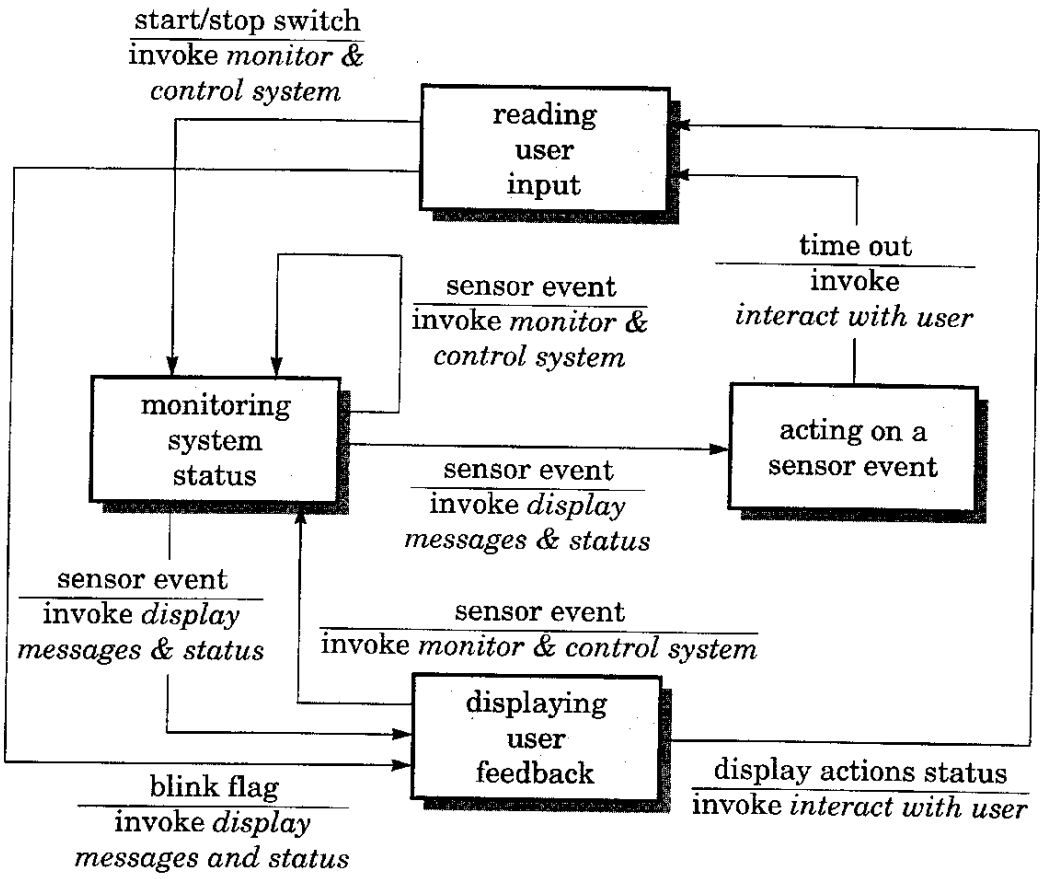
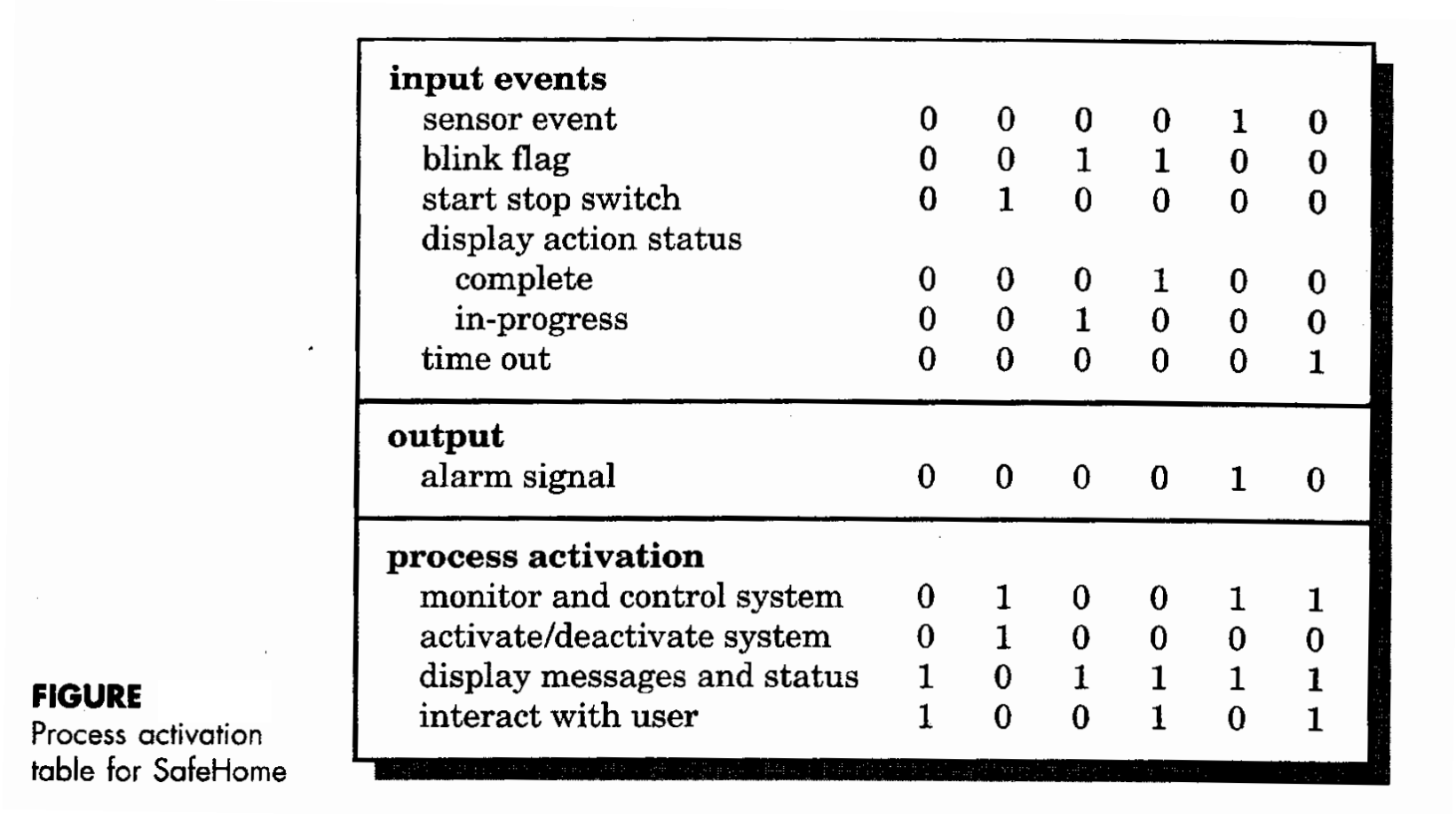


FIGURE
State-transition diagram for *SafeHome*

Process Activation Table



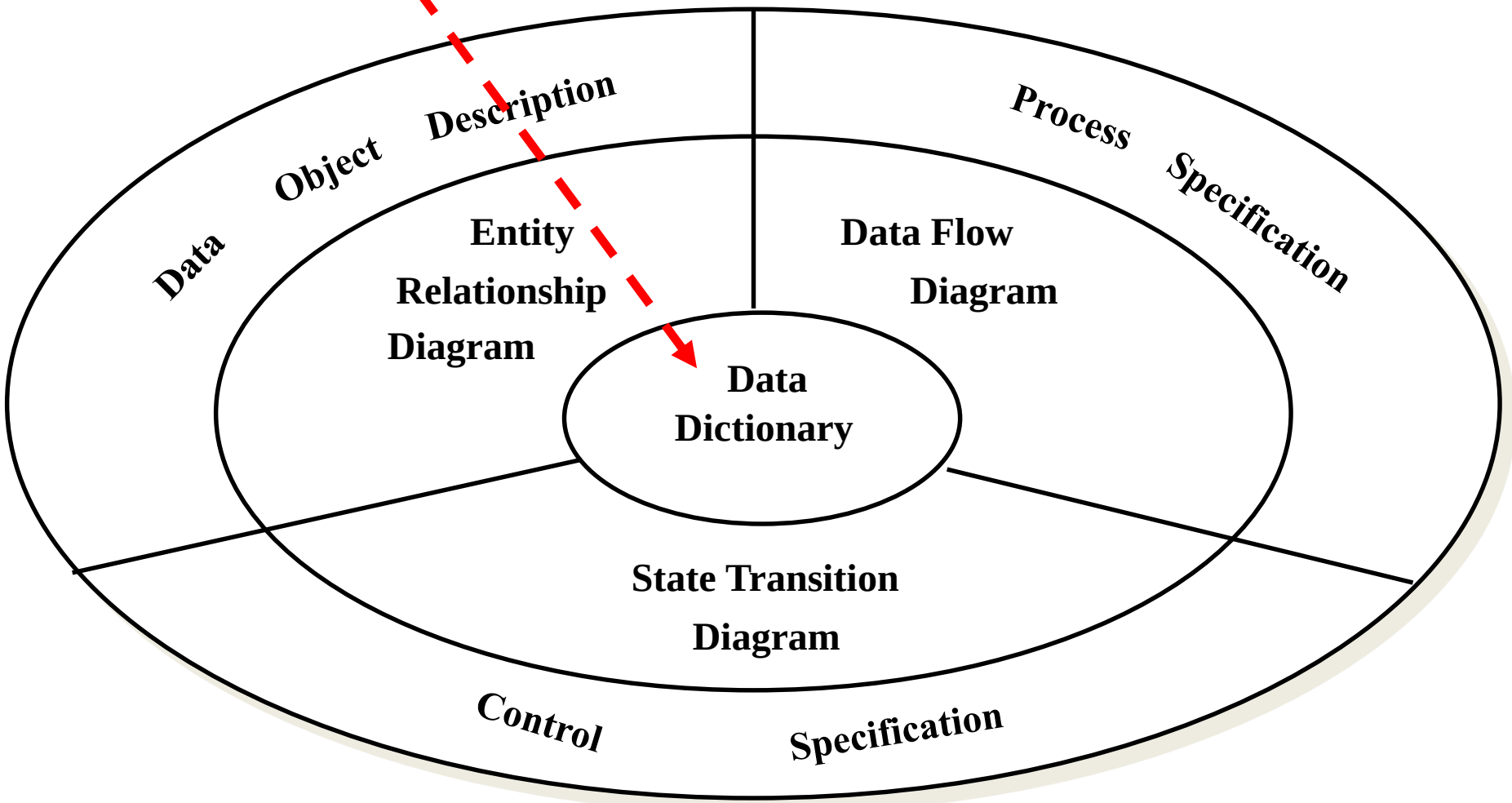
input events						
sensor event	0	0	0	0	1	0
blink flag	0	0	1	1	0	0
start stop switch	0	1	0	0	0	0
display action status						
complete	0	0	0	1	0	0
in-progress	0	0	1	0	0	0
time out	0	0	0	0	0	1
output						
alarm signal	0	0	0	0	1	0
process activation						
monitor and control system	0	1	0	0	1	1
activate/deactivate system	0	1	0	0	0	0
display messages and status	1	0	1	1	1	1
interact with user	1	0	0	1	0	1

FIGURE
Process activation table for SafeHome

Process Specifications

- Describes all flow model processes at final level of refinement:
  - Narrative text
  - Program design language description
  - Mathematical equations
  - Tables
  - Diagrams
  - Charts

Data Dictionary



Data Dictionary

- Why a data dictionary? Need an organized way to represent data & control characteristics
- Usual contents:
  - Name
  - Alias
  - Where and how used
  - Content description (of composite items)
  - Supplementary information, e.g., restrictions, limitations, preset values

Example

- Name: Shuttle pose
- Aliases: Position-orientation vector
- Where used: Display of Shuttle on map
- Content: x, y, z position wrt to Earth's Center, roll, pitch, yaw
- Supplementary Info: Elevation must be above 140 nautical miles

Data Dictionary

- Common tools supporting DD:
  - Preventing creation of duplicate names
  - Enforce naming conventions
  - Printing dictionary
  - Determine the range of impact of changes, i.e., which processes are affected
  - Assist configuration management

Summary

- Key elements
  - Data modeling:
    - Data objects, attributes and relationships
    - Cardinality and modality
    - Entity-relationship diagrams
  - Functional modeling:
    - Data and control flow diagrams
  - Behavioral modeling:
    - State transition diagrams
  - Data Dictionary