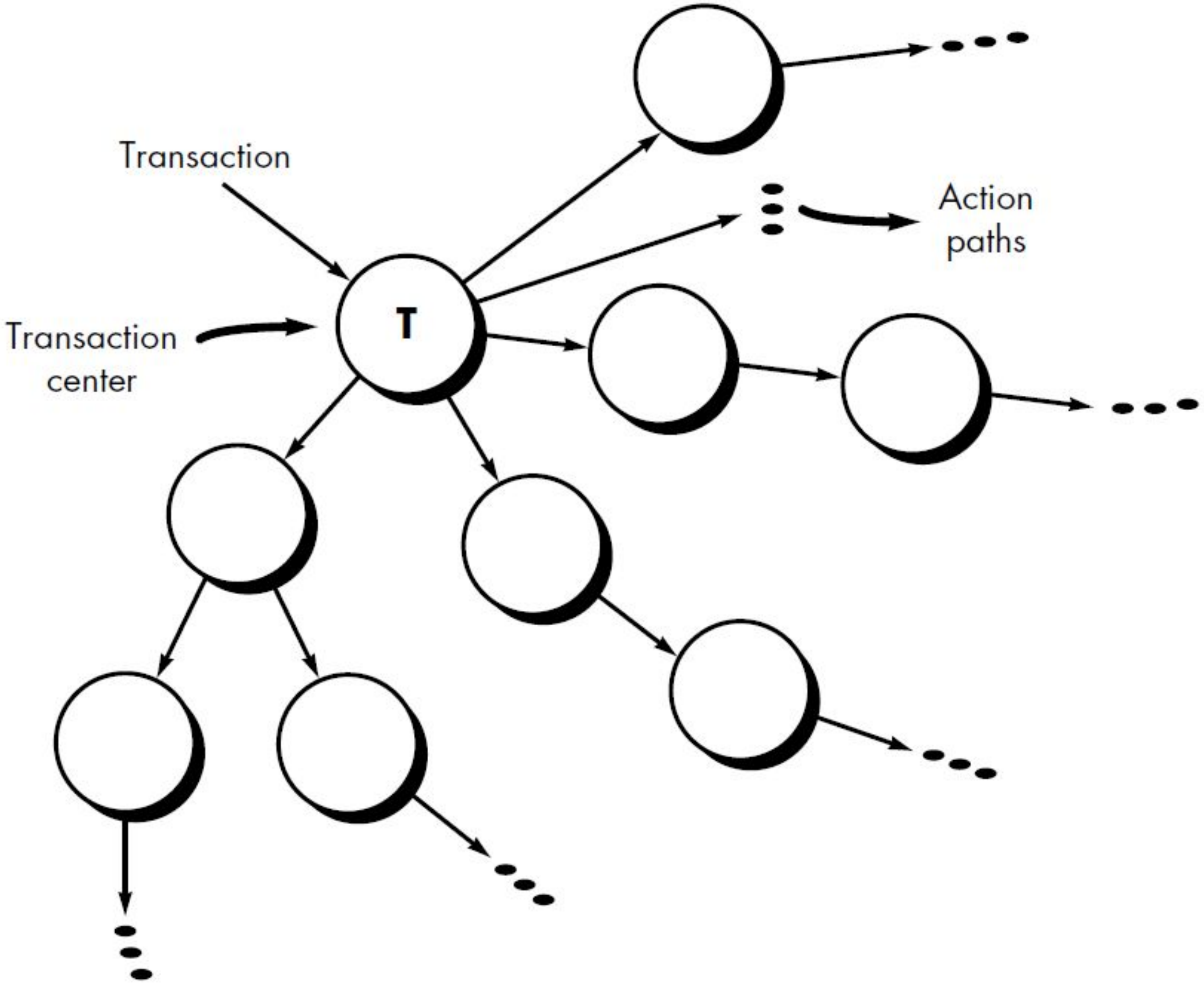
Architecture Design

Instructor: Mehroze Khan

Mapping Data Flow into Architecture

- A mapping technique, called ***structured design*** provides a convenient transition from a data flow diagram to software architecture
- Types of Information Flow:
  - **Transform Flow**
    - Flow of data occurs in a sequential manner and follows one, or only a few, "straight line" paths.
      - Incoming Flow: Conversion of data from external to internal form
      - Transform Center: Incoming data passed through Transform center
      - Outgoing Flow: Conversion from internal to external form
  - **Transaction Flow**
    - Single data item triggers data flows along one of many paths
      - Incoming Flow: Converts external world information into a transaction
      - Transaction Center: Hub of information flow from which many action paths come out
      - Action Paths: Transaction is evaluated and based on its value, flow along one of many *action paths* is initiated

Mapping Data Flow into Architecture
(Transaction Flow)



Mapping Data Flow into Architecture (Transform Mapping)

- Steps to map Data Flow Diagrams into a Software Architecture:

1. Review the fundamental system model
2. Review and refine data flow diagrams for the software
3. Determine whether DFD has transaction flow or transform flow characteristics
4. Isolate transform center by specifying incoming and outgoing flow boundaries
5. Perform “first-level factoring”
6. Perform “second-level factoring”
7. Refine the first-iteration architecture using design heuristics for improved software quality

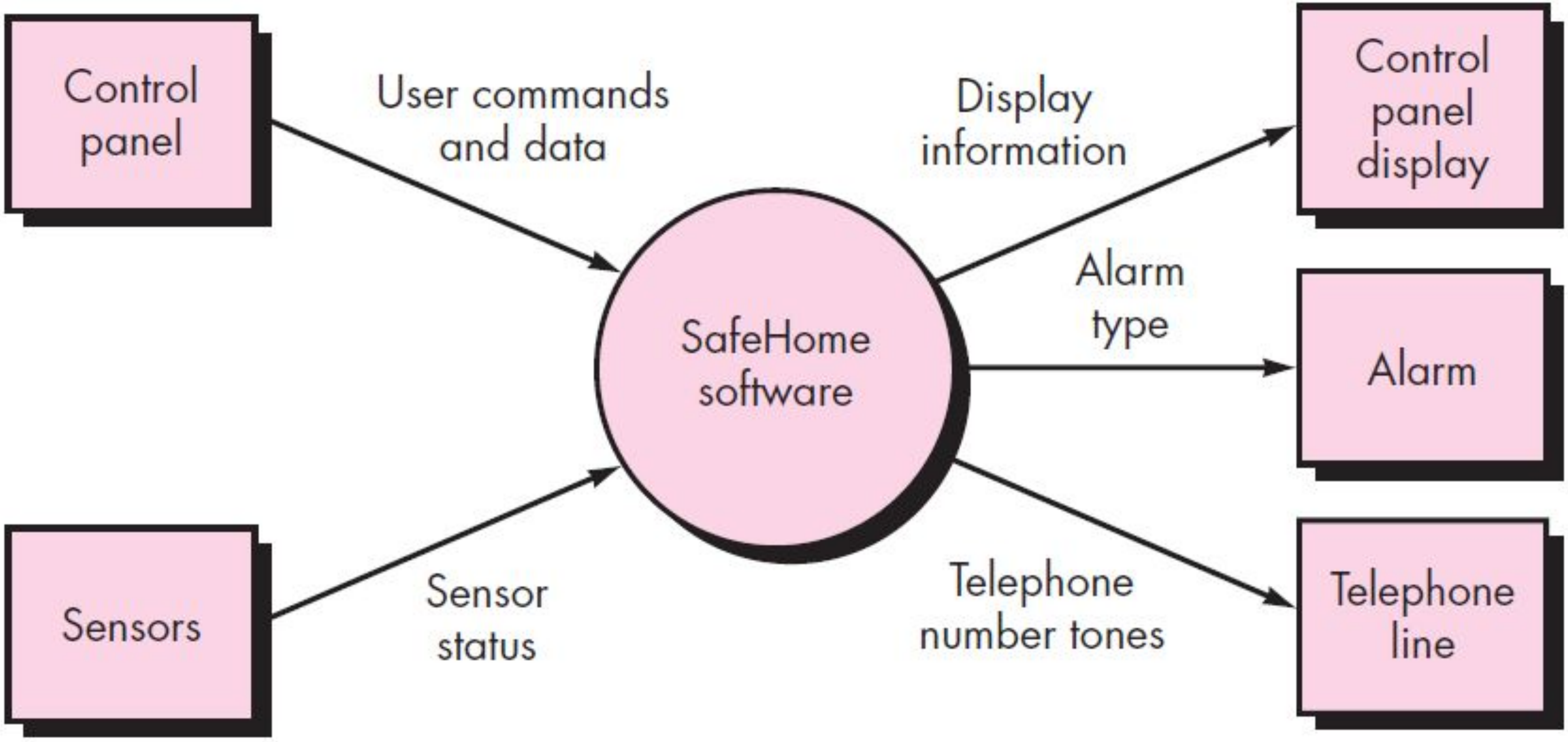
Example

- Our research indicates that the market for home security systems is growing at a rate of 40 percent per year.
- We would like to enter this market by building a microprocessor-based home security system that would protect against and/or recognize a variety of undesirable "situations" such as illegal entry, fire, flooding, and others.
- The product, tentatively called ***SafeHome****,* will use appropriate sensors to detect each situation, can be programmed by the homeowner, and will automatically telephone a monitoring agency when a situation is detected.

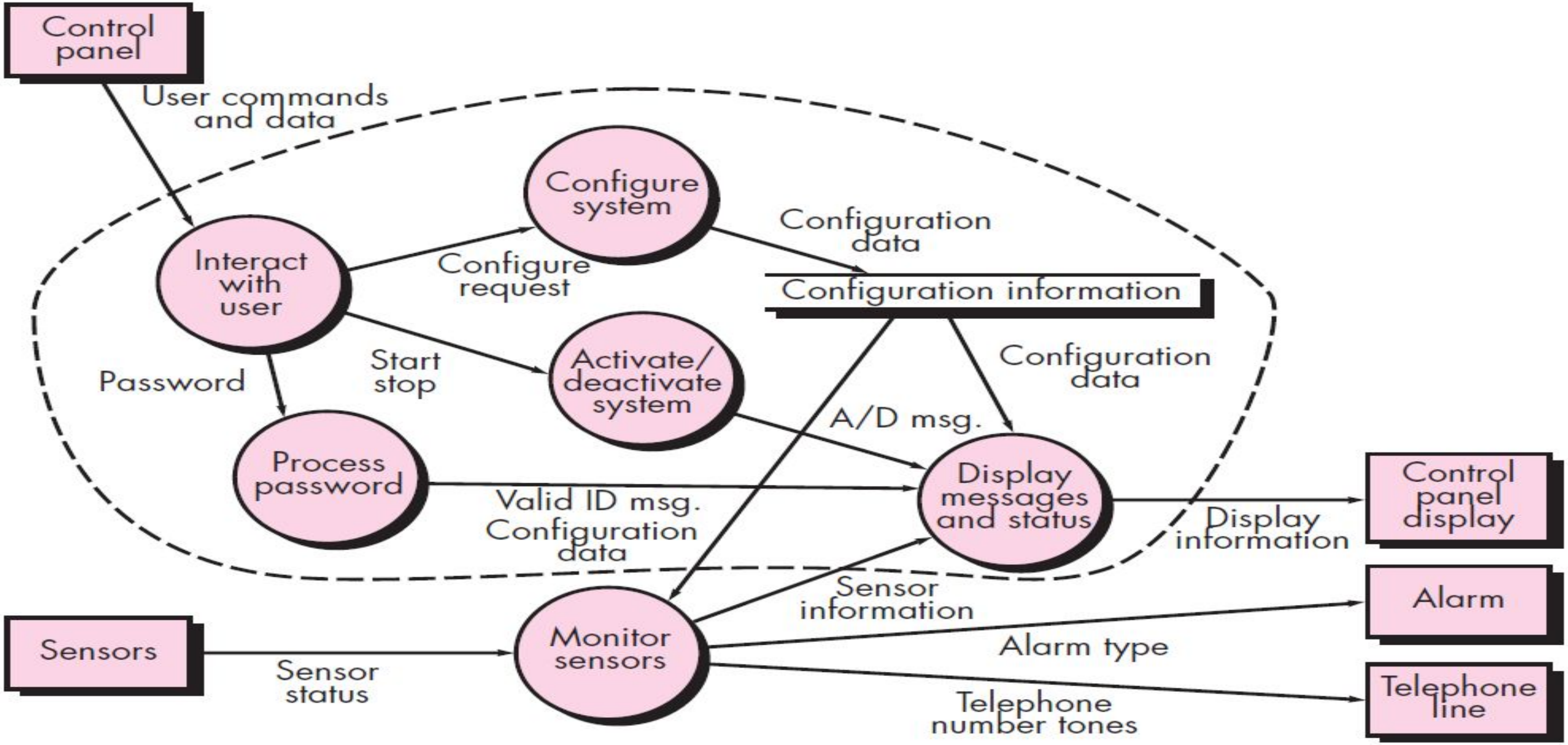
Example

Safe Home is a home management product that allows homeowners to control their homes using their PC and Cell phone remotely. Home security, home surveillance, appliance and device control are a few functionalities that are provided by the Safe Home product. Initial design tells that both the Internet interface and the other Graphical User Interface (GUI) are both part of external communication management module. Security related hardware management includes control panel processing (i.e., keypad and display function), detector management (i.e., scheduler and sensors etc.), alarm processing (i.e., phone communication, alarm management etc.). Other modules include Surveillance module and Home management module. Surveillance module monitors the inappropriate activities around and inside the home. Home management module allows the users to control devices (e.g., switch off A/C, tube light etc.).

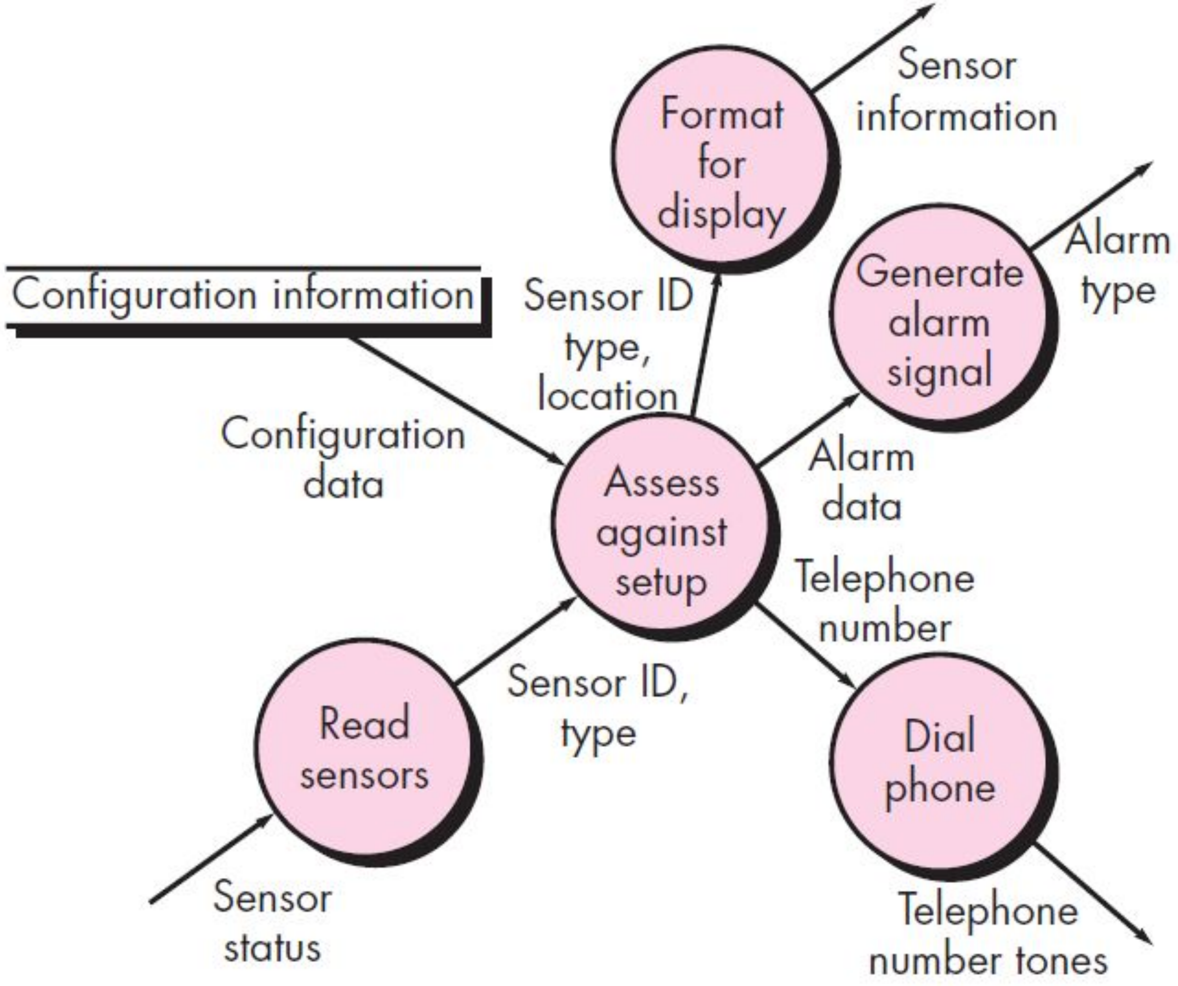
Step 1: Review the fundamental system model (Level-0 DFD)



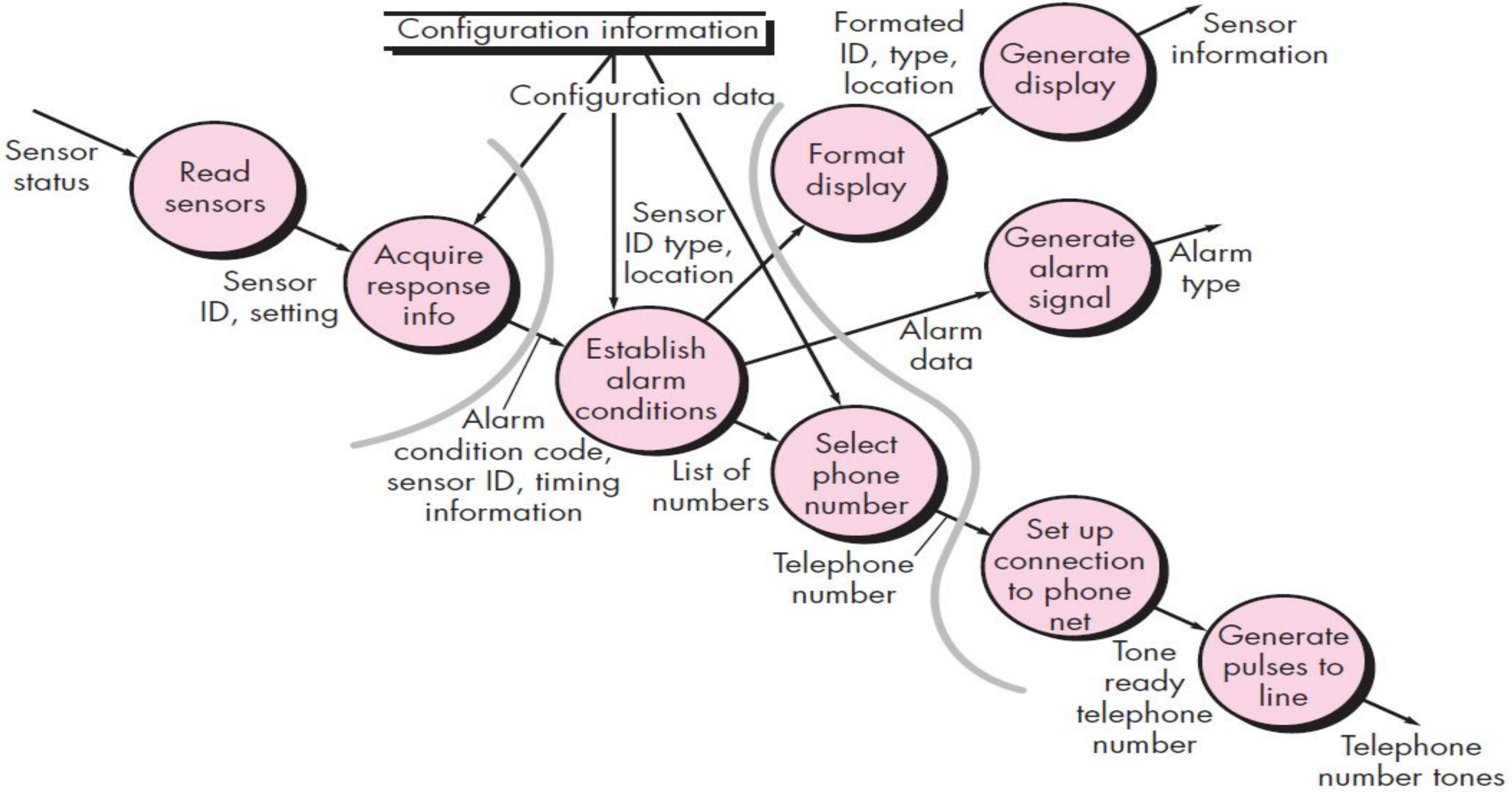
Step 2: Review and refine data flow diagrams for the software (Level-1 DFD for Monitor Sensors)



Step 2: Review and refine data flow diagrams for the software (Level-2 DFD for Monitor Sensors)



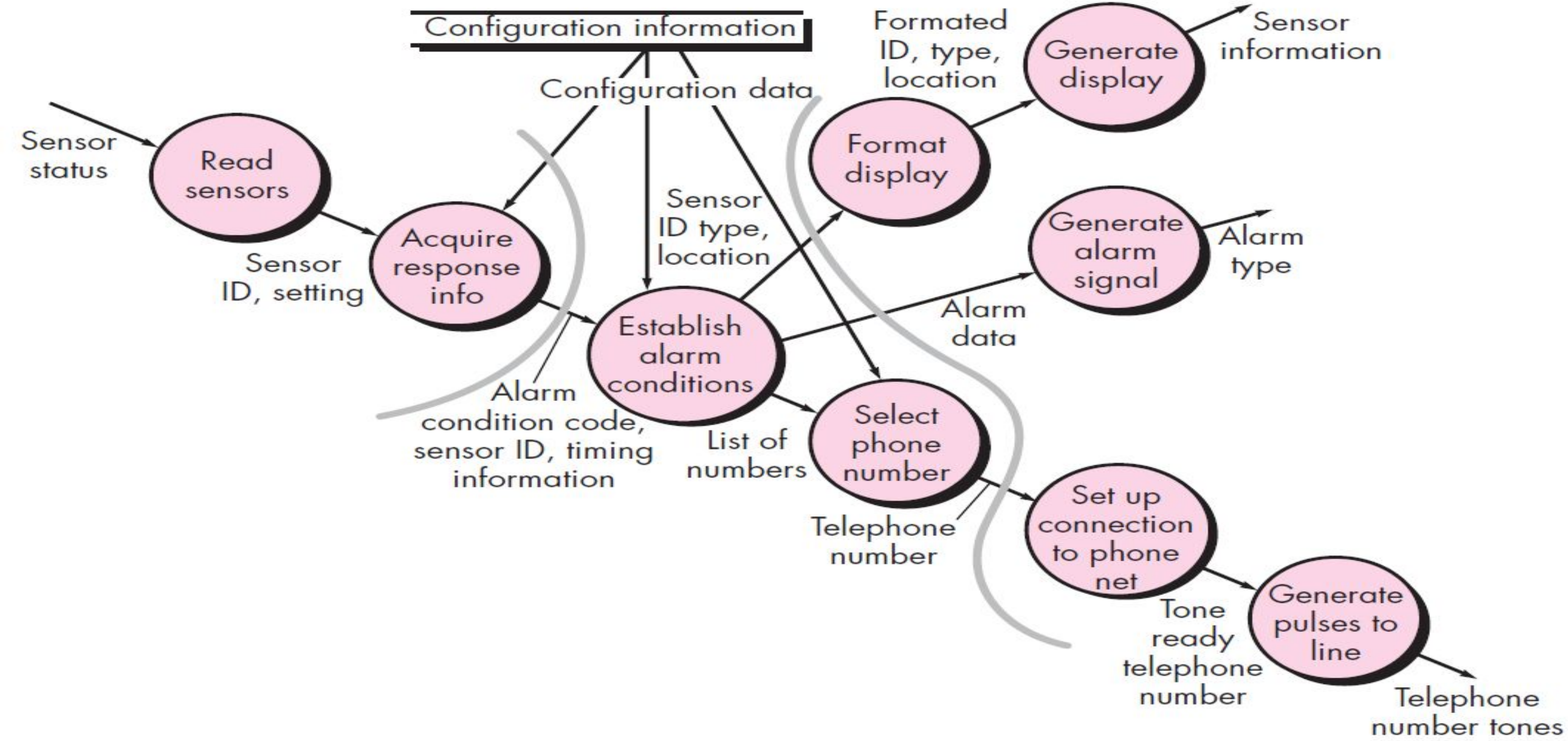
Step 2: Review and refine data flow diagrams for the software(Level-3 DFD)



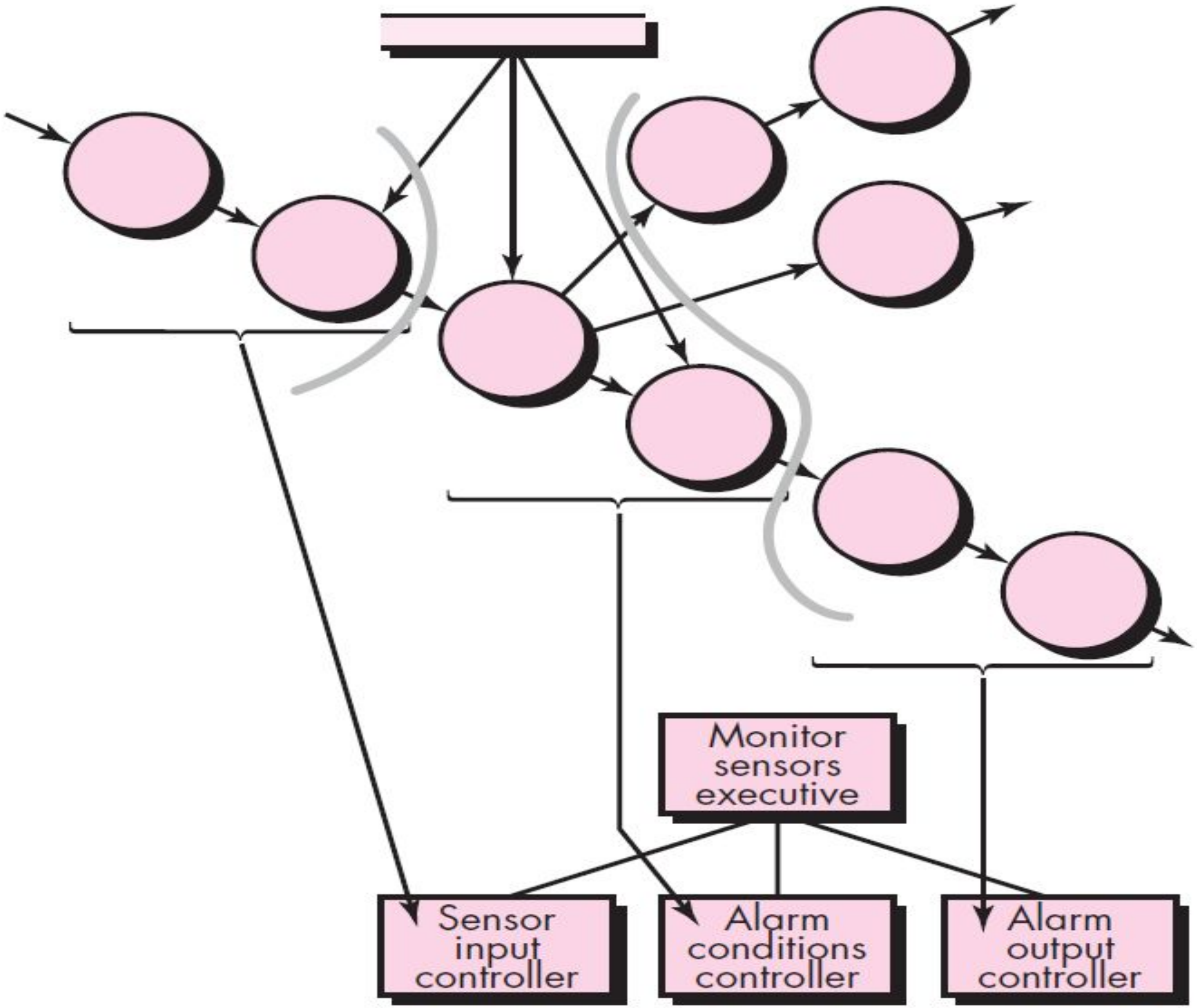
Step 3: Determine whether DFD has transaction flow or transform flow characteristics

- Evaluating the DFD, we see data entering the software along one incoming path and exiting along three outgoing paths. Therefore, an overall transform characteristic will be assumed for information flow.

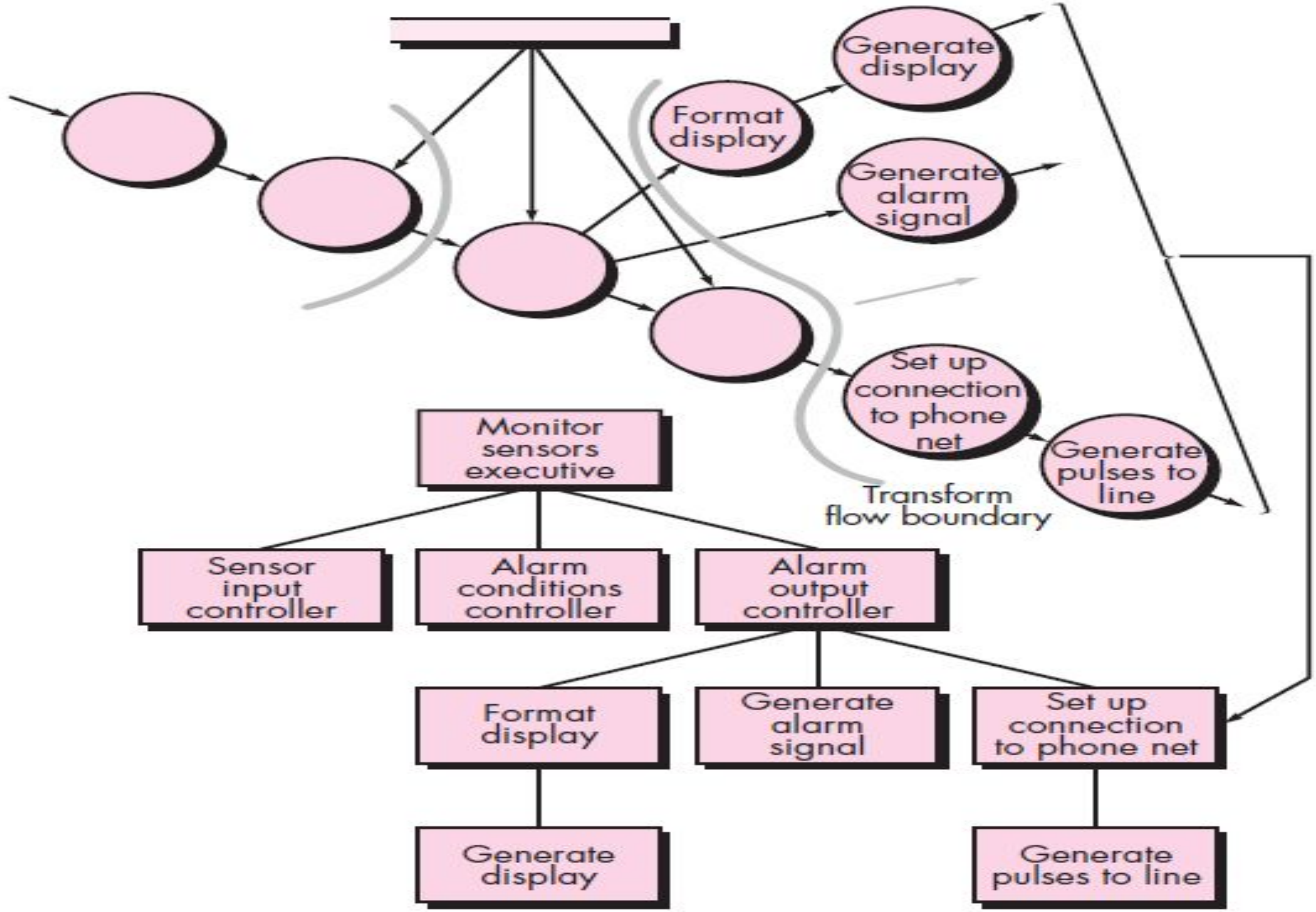
Step 4: Isolate transform center by specifying incoming and outgoing flow boundaries



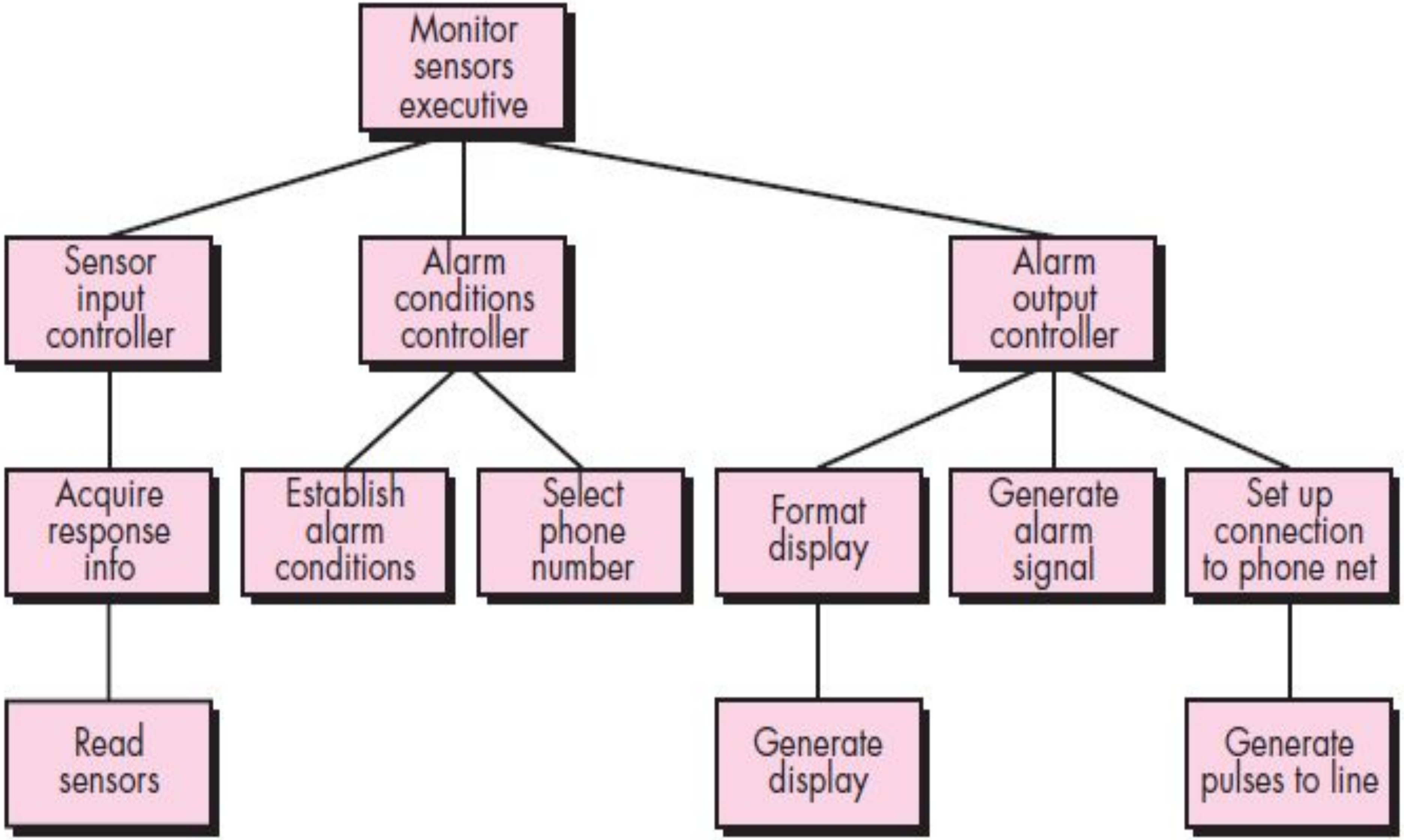
Step 5: Perform “first-level factoring”



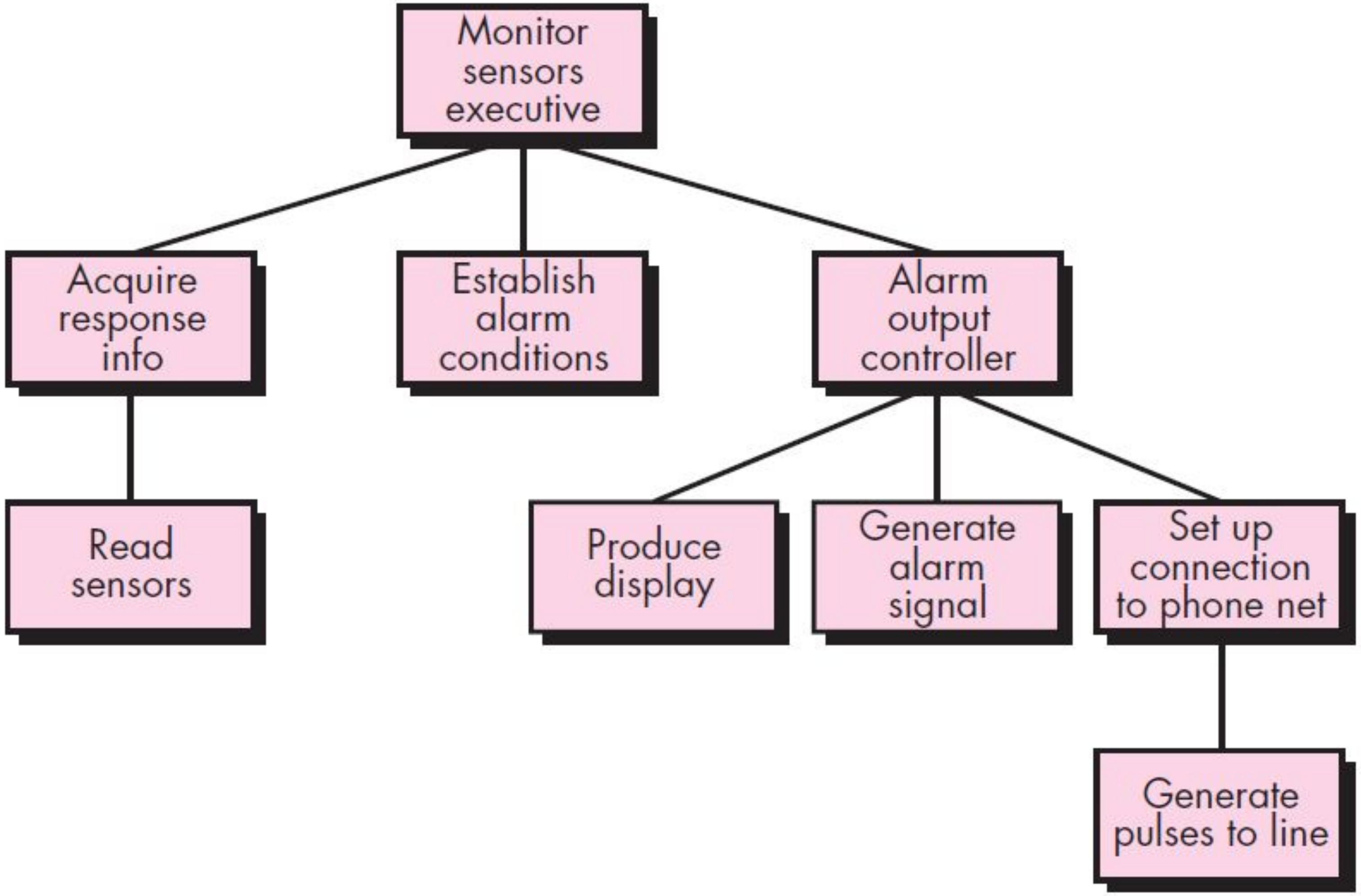
Step 6: Perform “second-level factoring”



Step 6: Perform “second-level factoring”



Step 7: Refine the first-iteration architecture using design heuristics for improved software quality

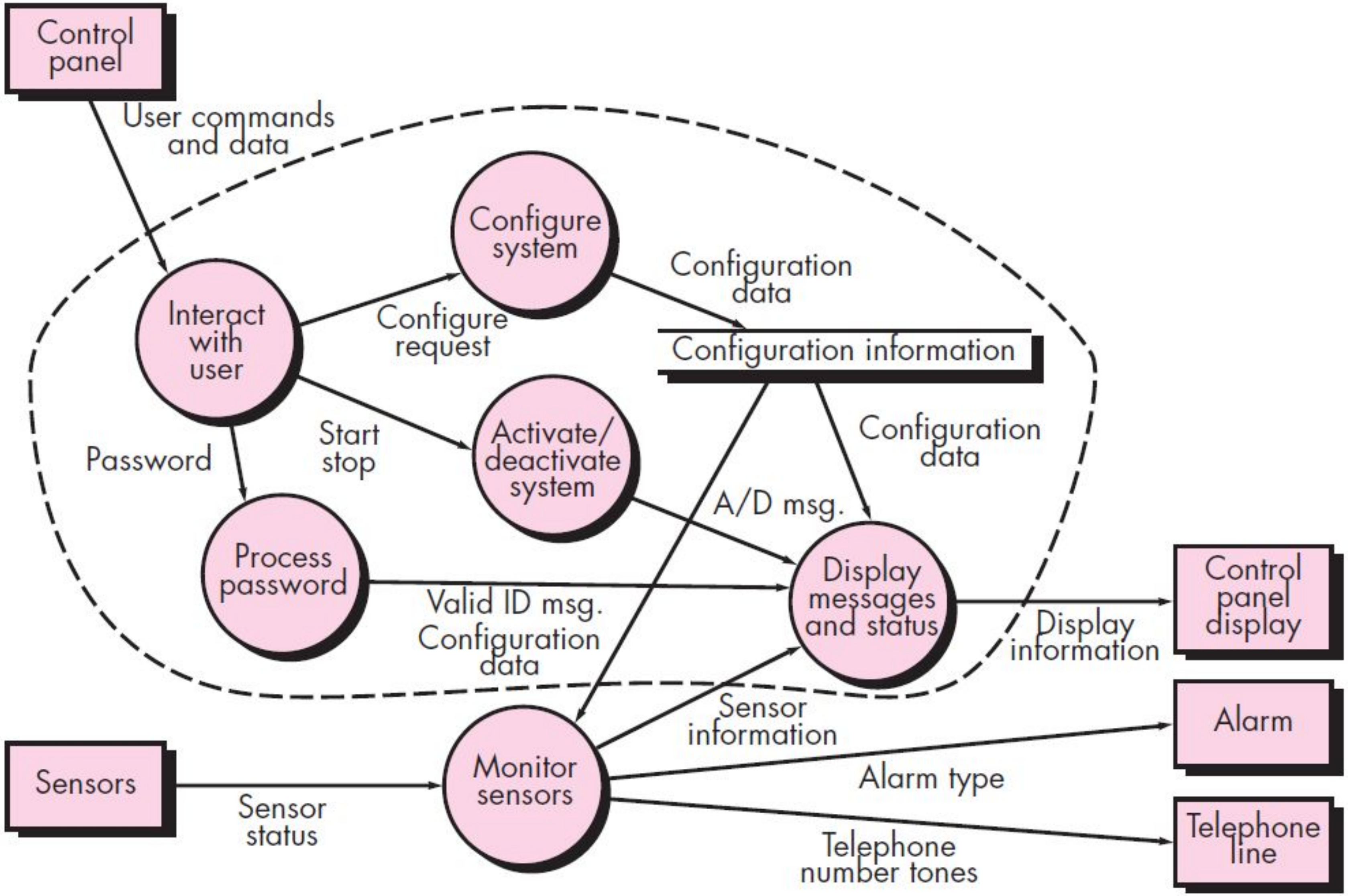


Mapping Data Flow into Architecture (Transaction Mapping)

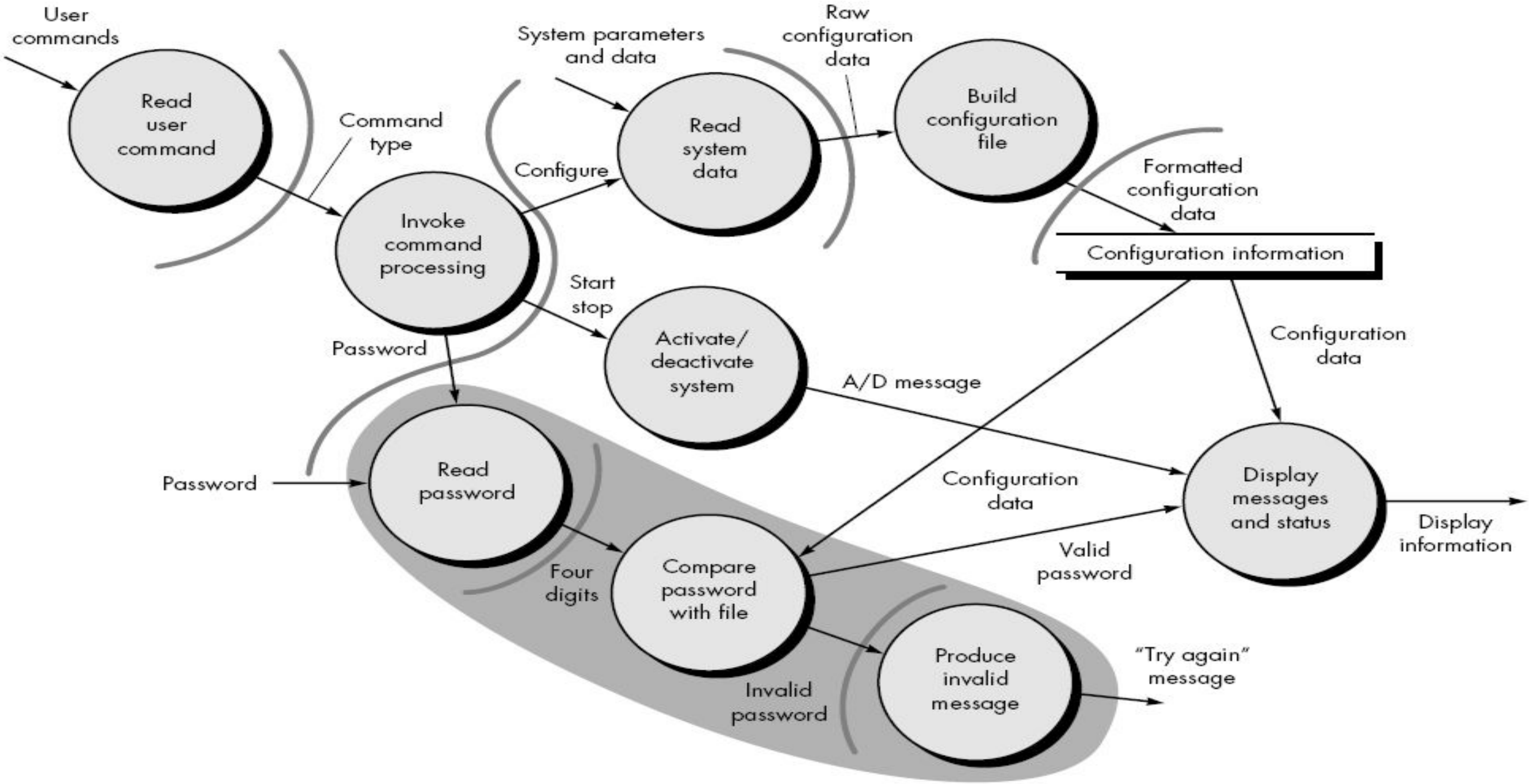
- Steps to map Data Flow Diagrams into a Software Architecture:

1. Review the fundamental system model.
2. Review and refine data flow diagrams for the software.
3. Determine whether the data flow diagram has transform flow or transaction flow characteristics.
4. Identify the transaction center and the flow characteristics along each of the action paths.
5. Map the data flow diagram in a program structure amenable to transaction processing.
6. Factor and refine the transaction structure and the structure of each action path.
7. Refine the first iteration architecture using design heuristics for improved software quality.

Level-1 DFD for SafeHome Security Function



Level 2 DFD for user interaction subsystem with flow boundaries



Step 3: Determine whether DFD has transaction flow or transform flow characteristics

- Steps 1, 2, and 3 are identical to corresponding steps in transform mapping.
- The DFD has a classic transaction flow characteristic. However, flow along two of the action paths emanating from the *invoke command processing* bubble appears to have transform flow characteristics. Therefore, flow boundaries must be established for both flow types.

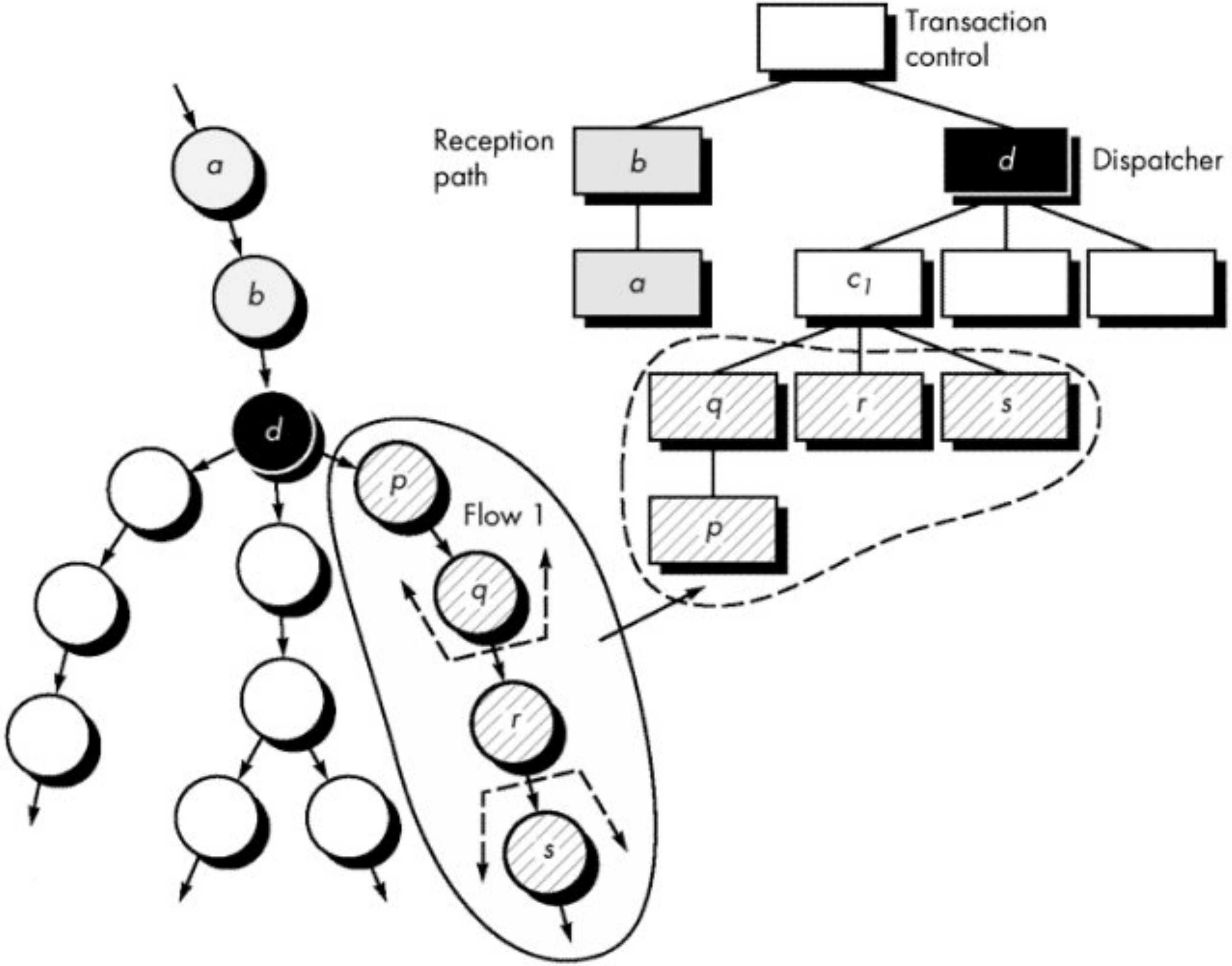
Step 4: Identify Transaction Center and flow characteristics along each of action paths

- The location of the transaction center can be immediately discerned from the DFD. The transaction center lies at the origin of a number of actions paths that flow radially from it. *Invoke command processing* bubble is the transaction center.
- The incoming path and all action paths must also be isolated.
- Each action path must be evaluated for its individual flow characteristic. For example, the "password" path (shown enclosed by a shaded area) has transform characteristics. Incoming, transform, and outgoing flow are indicated with boundaries.

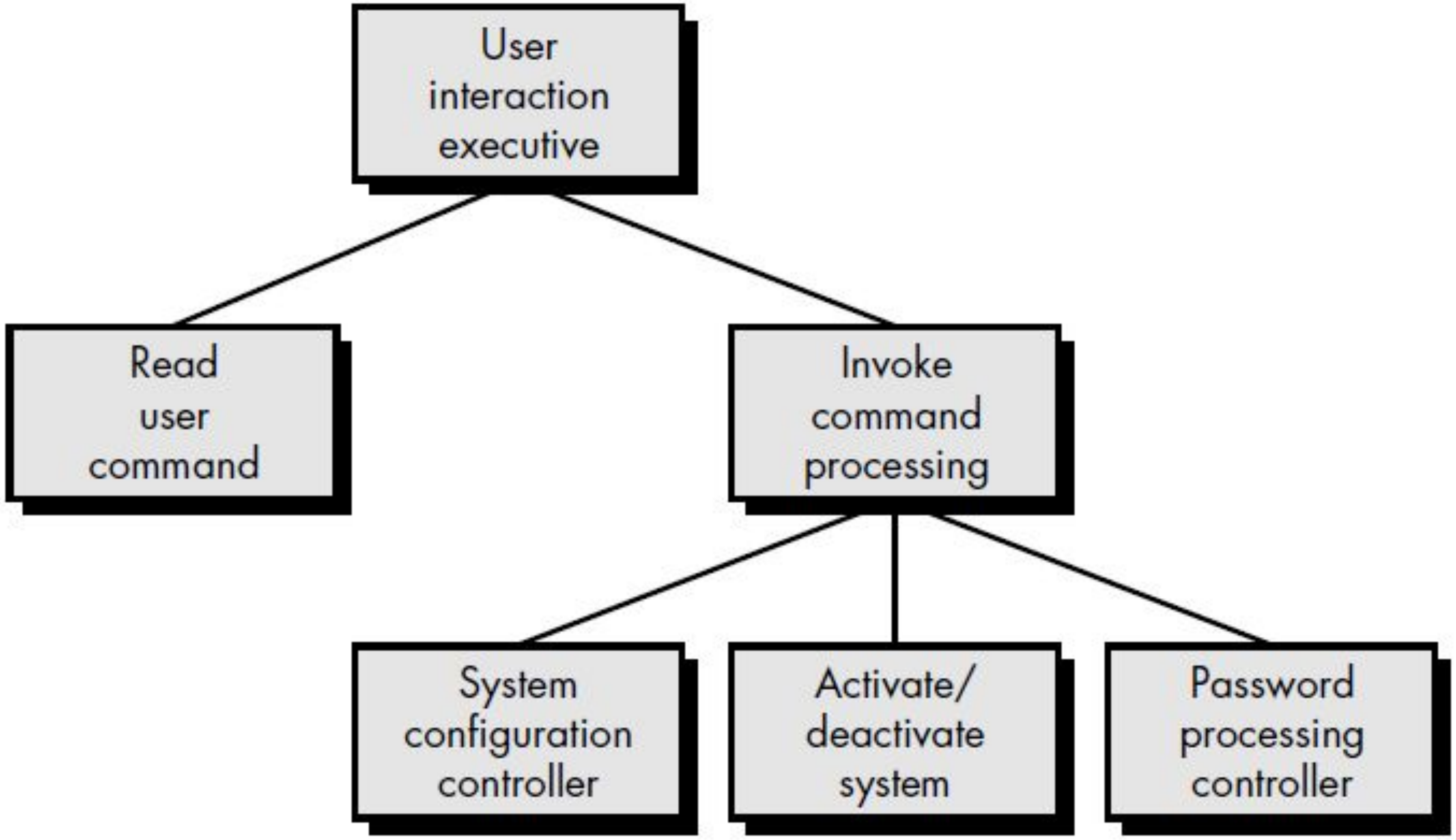
Step 5: Map the DFD in a program structure amenable to transaction processing

- Transaction flow is mapped into an architecture that contains an **incoming branch** and a **dispatch branch**.
- The structure of the incoming branch is developed in much the same way as transform mapping.
- Starting at the transaction center, bubbles along the incoming path are mapped into modules.
- The structure of the dispatch branch contains a dispatcher module that controls all subordinate action modules.

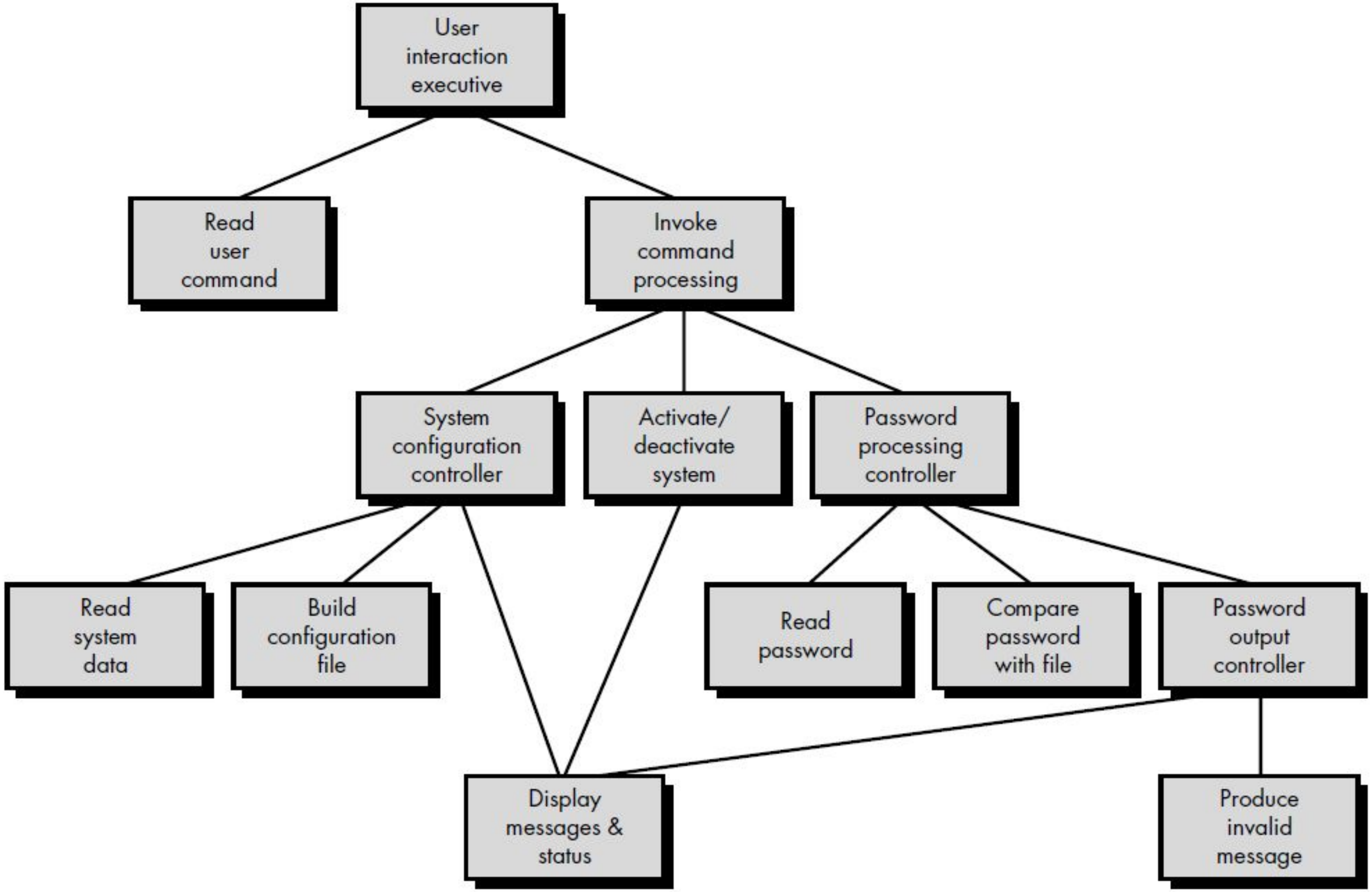
Transaction Mapping



First-level factoring for user interaction subsystem


User interaction executive
Read user command
Invoke command processing
System configuration controller
Activate/ deactivate system
Password processing controller

Step 6: Factor and refine the transaction structure and the structure of each action path


User interaction executive
Read user command
Invoke command processing
System configuration controller
Activate/ deactivate system
Password processing controller
Read system data
Build configuration file
Read password
Compare password with file
Password output controller
Display messages & status
Produce invalid message

Step 7: Refine the first architecture using design heuristics for improved software quality

- This step for transaction mapping is identical to the corresponding step for transform mapping.
- Criteria such as **module independence, practicality (efficacy of implementation and test), and maintainability** must be carefully considered.

References

- Roger S. Pressman, Software Engineering A Practitioner's Approach, 5th Edition. McGrawHill