Software Engineering I CSC-382

Lecture 16

Software Engineering I CS-382

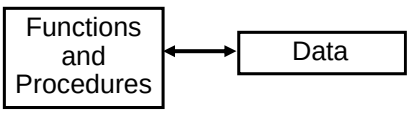
- Chapter 10
- What we will cover: (Architectural Design)
  - Second half of Chapter 10 in Pressman
  - Learn specifically how to perform an architectural decomposition and design.

Recall the Tools We Will Used for Analysis Modeling

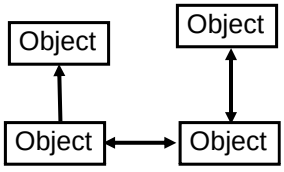
- **Structured Analysis-based Techniques**
  - Used Data Flow Diagramsfor the Flow Models and State Transition Diagrams for the Behavior Models
  - Finished with process narratives (PSPECS) to define low level requirements.
- **Object Oriented Analysis-based Techniques**
  - Derived the details of Class Models (started with CRC Cards) and their relationships (Class Diagrams), and behavior (sequence charts and State Diagrams)

Object Oriented Analysis vs. Structured Analysis

Structured Analysis Paradigm



Object-Oriented Analysis Paradigm

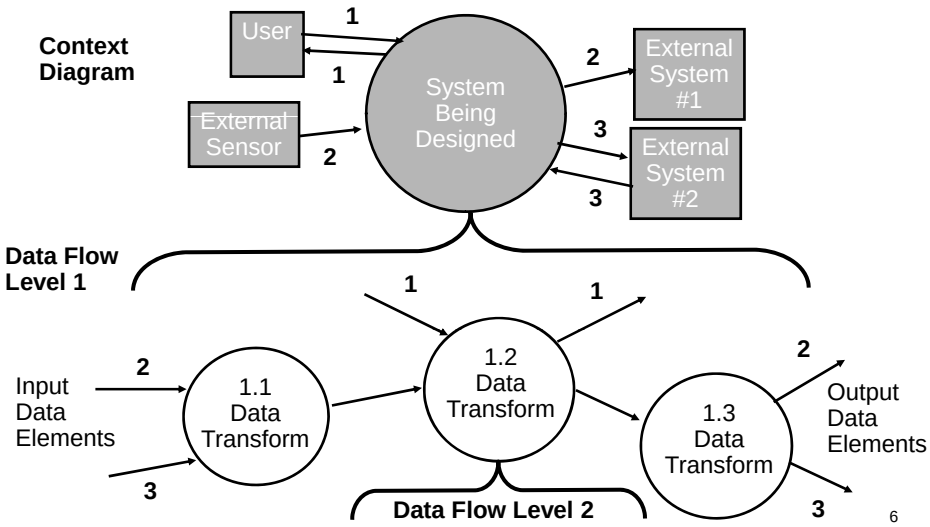


In the structured analysis model the data and the functions that operate on them are artificially separated.

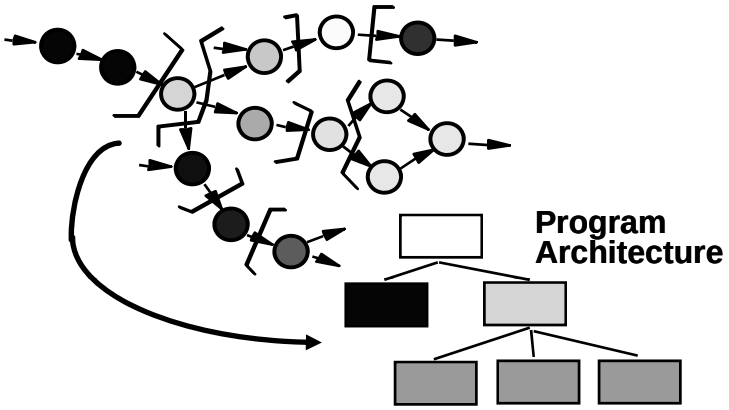
Recall Analysis Modeling via Structured Analysis

- Structured Analysis viewed a system as a sequence of transformations operating on the input data and leading to the final outputs
  - Represents how data objects are transformed at they move through the system
  - Starts with the Context Diagram
  - Continue with Data Flow Diagrams to define all the data transformations
  - Down to the lowest level uses Process Specifications (PSPECS) (narratives) to define the data transformations required at the lowest desired level of modeling

Data Flow Diagram Process



Deriving Program Architecture



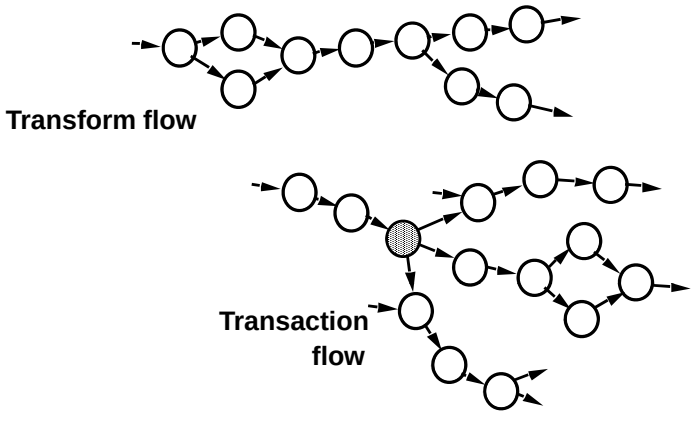
Steps for Structured Analysis Based Architectural Design

1. Review the system model
   - Be sure all I/O well defined in the software context diagram
2. Refine the data flows
   - Recall we stopped at a level we felt defined the requirements
   - Now we will continue that and re-write some of the PSPECS as continued data flows

Steps for Structured Analysis Based Architectural Design II

3. Analyze flows for **transactions** versus simple **transforms**
   - Transactions imply a distribution of information along multiple paths and thus a different control and communication architecture

Types of DFD Flows



Steps for Structured Analysis Based Architectural Design III – for the transforms

4. Define the transform center
   - This will define core processing from input and output formatting and processing
   - These will then lead to groupings of transforms that naturally turn into system-level components

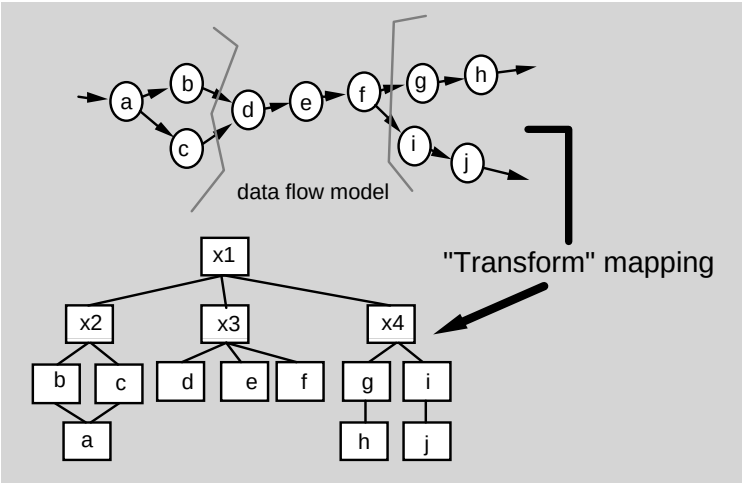
Steps for Structured Analysis Based Architectural Design IV – for the transforms

5. Perform 1st level factoring
   - The basic idea is to devise the highest level groupings of your system's data transforms
     - We should really try a few groupings at this level each with a different architecture in mind.
     - Book assumes a call-return architecture right away, but you can develop this top level view with any architecture in mind (recall the pipes and filters, etc.) -
     - Decide where you may need some missing control modules

Steps for Structured Analysis Based Architectural Design V – for the transforms

6. Perform 2nd level factoring
   - Starting at the transform center collect together sets of transforms that have some natural cohesion.
     - Recall the definitions for types of cohesion and group at the highest levels possible.
7. Refine using all the rules for software quality
   - Ensure good cohesiveness, minimize coupling, etc.

Transform Mapping



Steps for Structured Analysis Based Architectural Design VI – for the transactions

4. Define the transaction center
   - Define the characteristics of the flows along each path
   - Each path out of a transaction center will then lead to groupings of transforms that naturally turn into other system-level components

Steps for Structured Analysis Based Architectural Design VII – for the transactions

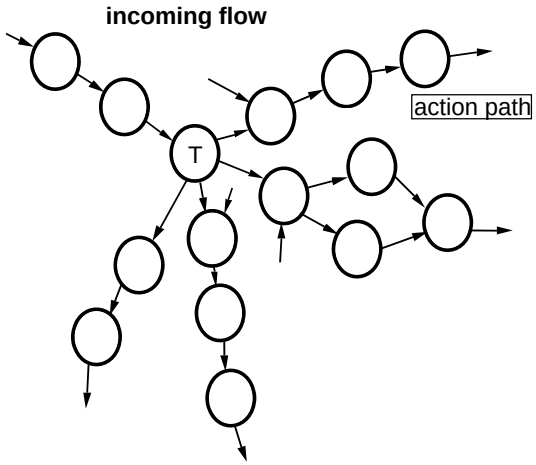
5. Perform 1st level factoring (Book calls it map DFDs…)
   - Work “upstream” from the transaction point.
   - Group all the transforms that lead up to it
   - Define the architecture of the actual transaction point, as to how the flow is controlled and communicated down each subsequent processing path

Steps for Structured Analysis Based Architectural Design VIII– for the transactions

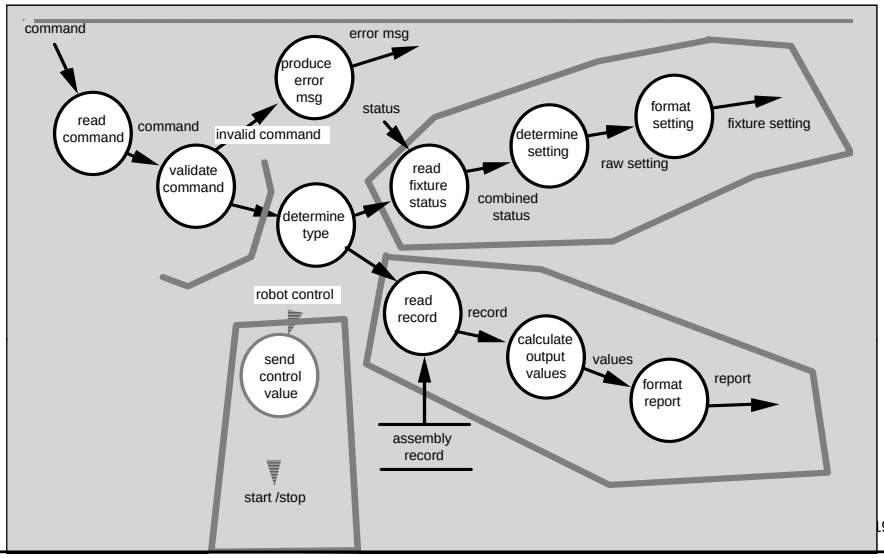
6. Perform 2nd level factoring of each transaction path
   - Now work “down-stream” from the transaction point
   - Starting at the transaction point and define appropriate structure for each path.

7. Refine using all the rules for software quality
   - Ensure good cohesiveness, minimize coupling, etc.

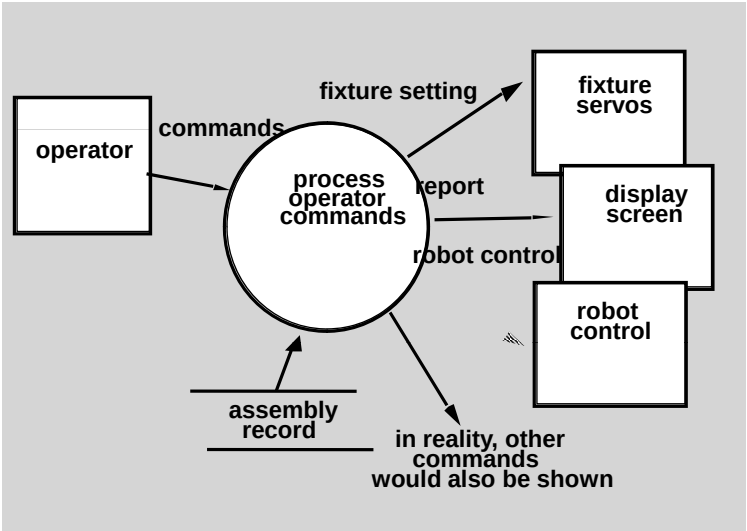
Transaction Flow



Isolate Flow Paths

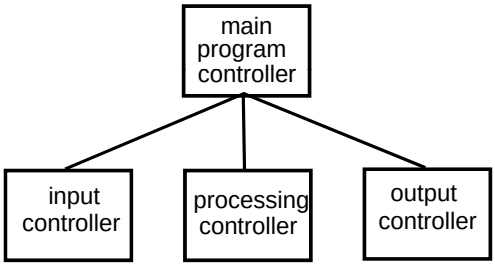


Transaction Example



Overview of Structured Method

- Start at processing center and work way out.
- Simplest architecture is then input, processing, output
  - Note Structured method drives the top down control
  - This is exactly the approach we teach in 175 for developing a program



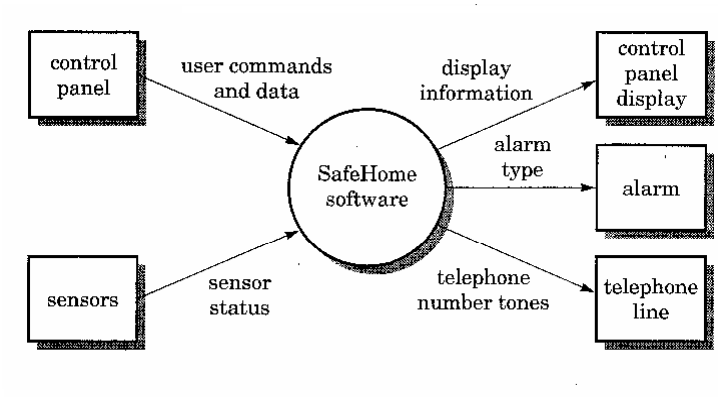
Overview of Structured Method

- We continue the refinement after this first level
  - The central transform is treated like the top level system and it is divided into a similar set of three
  - Likewise when factor the input or output subsystem, treat them like a central transform and divide until reach outer interfaces
  - Use our rules for cohesion and coupling to guide us

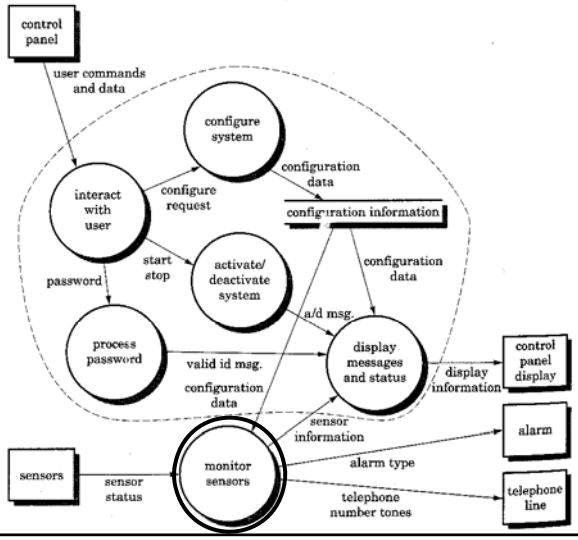
Overview of Structured Method II

- The traditional SD approaches assume the higher level structures are coordinators or controllers
  - The majority of the work is performed at the lower levels
  - System is fully factored when all the actual processing is accomplished by the bottom-level atomic modules
  - The higher level non-atomic modules perform control and coordination

Structured Method for Safehome



Structured Method for Safehome – top level DFD



Structured Method for Safehome – flow down for monitor sensors

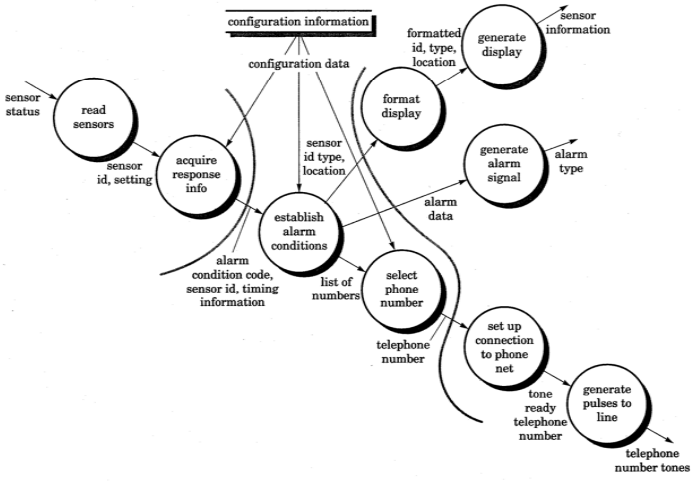


FIGURE 14.6. Level 3 DFD for *monitor sensors* with flow boundaries

Structured Method for Safehome – partition into the three components

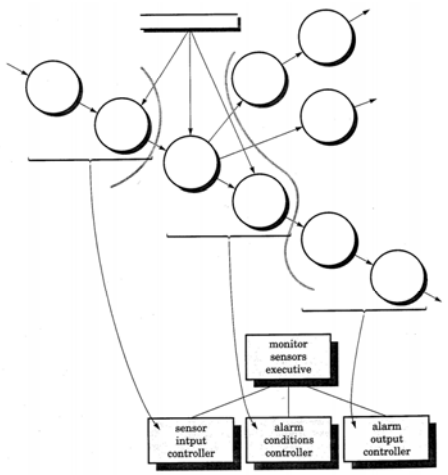


FIGURE 14.7. First-level factoring for *monitor sensors*

Structured Method for Safehome – next refinement of structure

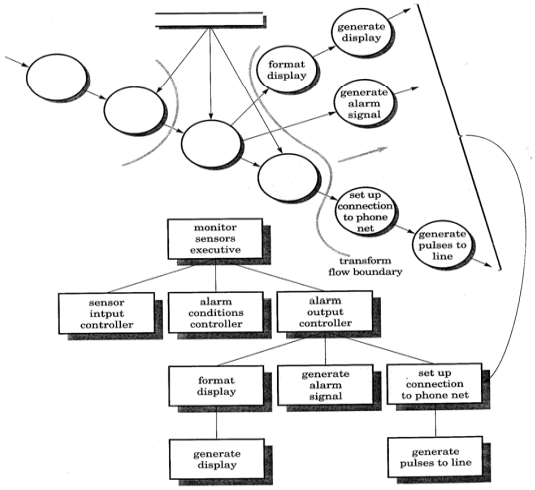


FIGURE 14.8. First level factoring for *monitor sensors*

Summary of Structured Method

- "It is essentially a top-down functional refinement technique that identifies the hierarchy of modules.." – Jalote
- Is this really the best way to address many complex systems though ?
  - Particularly distributed systems do not necessarily follow this strict hierarchy
  - Also we learned so many interesting architectures besides the call and return architecture

Summary of Structured Method II

- My advice is to use the modeling technique thru to the DFDs
  - Then use our architectural models and decide how to partition your DFD across one of these architectures
  - This is not the traditional way, but it allows Structured Methods to be integrated with more modern ideas of architecture

What Concepts Can We Use For Helping Us Define Architecture?

- Two key concepts to recall are Cohesion and Coupling
- Cohesion measures?
  - How single minded and focused the components are
- Coupling measures?
  - How much interconnection there is between the components
- When we try to divide our system into components we should review the possible components against these concepts
  - Try to come up with a few options and you can compare them qualitatively based on these concepts

For Next Class

- Chapter 10 – OO-based Architectural Modeling