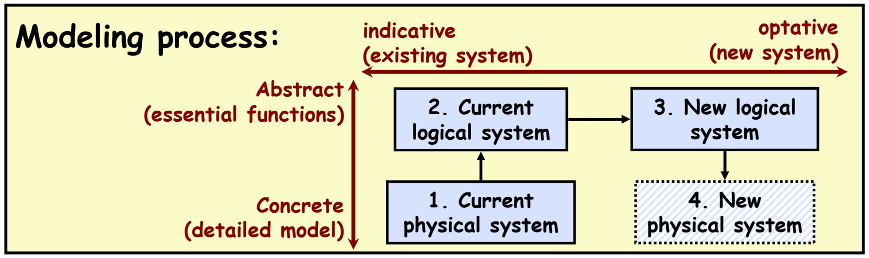
Lecture 15:
Structured Modeling Methods

- → Basics of Structured Analysis
  - ✎ Notations used
  - ✎ Modeling Process
- → Variants
  - ✎ SADT
  - ✎ SASS
  - ✎ SSADM
  - ✎ SRD
- → Advantages and Disadvantages

Structured Analysis

- → Definition
  - ✎ Structured Analysis is a data-oriented approach to conceptual modeling
  - ✎ Common feature is the centrality of the dataflow diagram
  - ✎ Mainly used for information systems
    - ➢ variants have been adapted for real-time systems



Modeling process:

indicative (existing system)

optative (new system)

Abstract (essential functions)

Concrete (detailed model)

2. Current logical system

3. New logical system

1. Current physical system

4. New physical system

- ✎ Model of current physical system only useful as basis for the logical model
- ✎ Distinction between indicative and optative models is very important:
  - ➢ Must understand which requirements are needed to continue current functionality, and which are new with the updated system

Central Concepts

Source: Adapted from Svoboda, 1990, p257

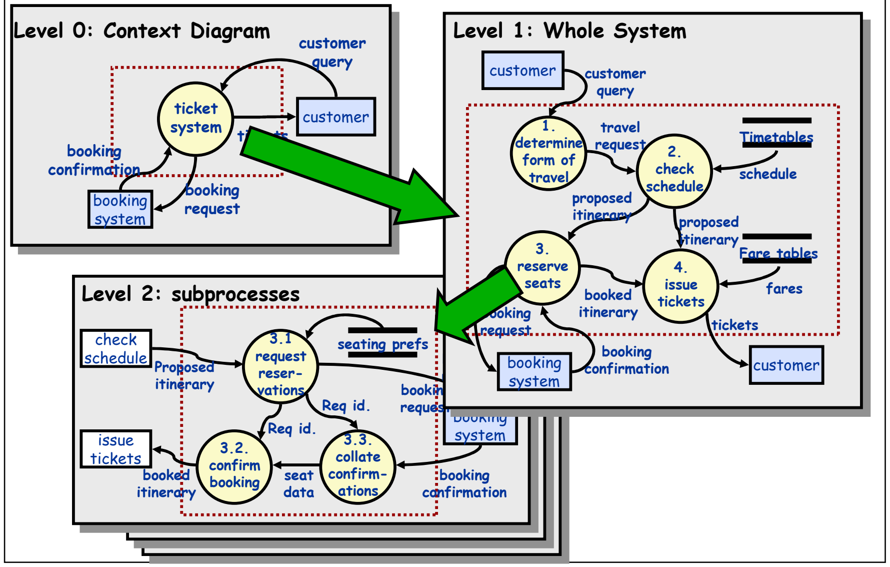
- → Process (data transformation)
  - ✎ activities that transform data
  - ✎ related by dataflows to other processes, data store, and external entities.
- → Data flow
  - ✎ indicate passage of data from output of one entitie to input of another
  - ✎ represent a data group or data element
- → Data store
  - ✎ a place where data is held for later use
  - ✎ Data stores are passive: no transformations are performed on the data
- → External entity
  - ✎ An activity outside the target system
  - ✎ Acts as source or destination for dataflows that cross the system boundary
  - ✎ External entities cannot interact directly with data stores
- → Data group
  - ✎ A cluster of data represented as a single dataflow
  - ✎ Consists of lower level data groups, or individual elements
- → Data element
  - ✎ a basic unit of data

Modeling tools

Source: Adapted from Svoboda, 1990, p258-263

- → Data flow diagram
  - ✎ Context diagram ("Level 0")
    - ➢ whole system as a single process
  - ✎ intermediate level DFDs decompose each process
  - ✎ functional primitives are processes that cannot be decomposed further
- → Data dictionary
  - ✎ Defines each data element and data group
  - ✎ Use of BNF to define structure of data groups
- → Primitive Process Specification
  - ✎ Each functional primitive has a "mini-spec"
  - ✎ these define its essential procedural steps
  - ✎ Expressed in English narrative, or some form of pseudo-code
- → Structured Walkthrough

Hierarchies of DFDs



Data Dictionary & Process Specs

Source: *Adapted from Svoboda, 1990, p262-4*

Example Data Dictionary

Mailing Label =
  customer_name +
  customer_address

customer_name =
  customer_last_name +
  customer_first_name +
  customer_middle_initial

customer_address =
  local_address +
  community_address + zip_code

local_address =
  house_number + street_name +
  (apt_number)

community address =
  city_name + [state_name |
  province_name]

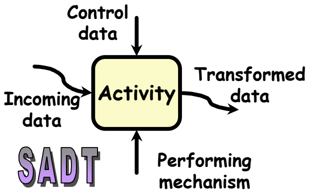
Example Mini-Spec

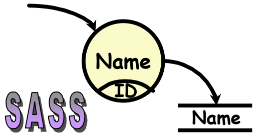
```
FOR EACH Shipped-order-detail
    GET customer-name + customer-address
    FOR EACH part-shipped
        GET retail-price
        MULTIPLY retail-price by
            quantity-shipped
        TO OBTAIN total-this-order
CALCULATE shipping-and-handling
ADD shipping-and-handling TO
    total-this-order
    TO OBTAIN total-this-invoice
PRINT invoice
```

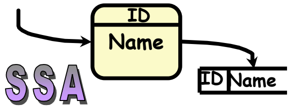
DFD variants

Source: *Adapted from Svoboda, 1990, p264-5*

- Structured Analysis and Design Technique (SADT)
  - Developed by Doug Ross in the mid-70's
  - Uses activity diagrams rather than dataflow diagrams
  - Distinguishes control data from processing data
- Structured Analysis and System Specification (SASS)
  - Developed by Yourdon and DeMarco in the mid-70's
  - 'classic' structured analysis
- Structured System Analysis (SSA)
  - Developed by Gane and Sarson
  - Notational style slightly different from Yourdon & DeMarco
  - Adds data access diagrams to describe contents of data stores
- Structured Requirements Definition (SRD)
  - Developed by Ken Orr in the mid-70's
  - Introduces the idea of building separate models for each perspective and then merging them







SASS methodology

Source: *Adapted from Davis, 1990, p83-86*

1. Study current environment

- draw DFD to show how data flows through current organization
- label bubbles with names of organizational units or individuals

2. Derive logical equivalents

- replace names with action verbs
- merge bubbles that show the same logical function
- delete bubbles that don't transform data

3. Model new logical system

- Modify current logical DFD to show how info will flow once new system is in place
- Don't distinguish (yet) which components will be automated

4. Define a number of automation alternatives

- document each as a physical DFD
- Analyze each with cost/benefit trade-off
- Select one for implementation
- Write the specification

Alternative Process Model: SRD

Source: *Adapted from Davis, 1990, p72-75*

1. Define a user-level DFD

- interview each relevant individual in the current organization
  - actually a role, rather than an individual
- Identify the inputs and outputs for that individual
- Draw an 'entity diagram' showing these inputs and outputs

2. Define a combined user-level DFD

- Merge all alike bubbles to create a single diagram
- Resolve inconsistencies between perspective

3. Define the application-level DFD

- Draw the system boundary on the combined user-level DFD
- Then collapse everything within the boundary into a single process

4. Define the application-level functions

- label the inputs and outputs to show the order of processing for each function
  - I.e. for function A, label the flows that take part in A as A1, A2, A3,...

Later developments

Later work recognized that:

- development of both current physical and current logical models is overkill
- top down development doesn't always work well for complex systems
- entity-relationship diagrams are useful for capturing complex data

Structured Analysis / Real Time (SA/RT)

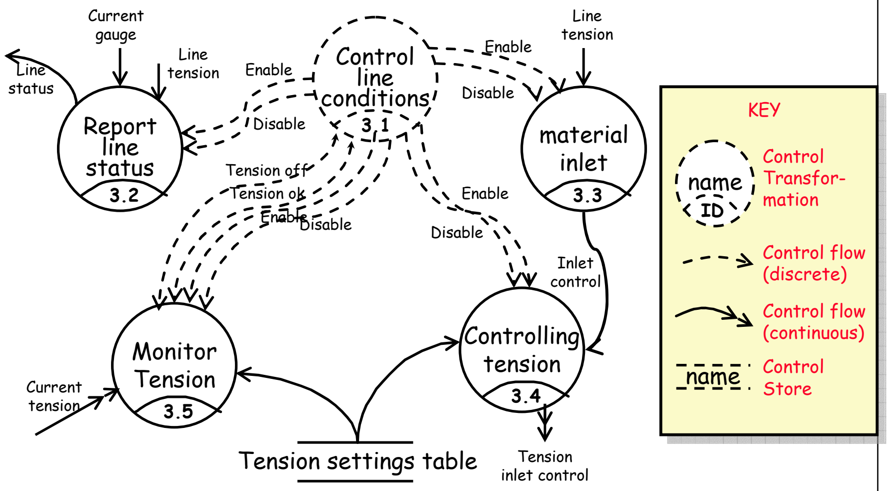
- Developed by Ward and Mellor in the mid-80's
- Extends structured analysis for real-time systems
  - Adds control flow, state diagrams, and entity-relationship models

Modern Structured Analysis

- Captured by Yourdon in his 1989 book
- Uses two models: the environmental model and the behavioral model
  - together these comprise the *essential* model
- Includes plenty of advice culled from many years experience with structured analysis

Real-time extensions

Source: *Adapted from Svoboda, 1990, p269*



Evaluation of SA techniques

Source: *Adapted from Davis, 1990, p174*

Advantages

- Facilitate communication.
- Notations are easy to learn, and don't require software expertise
- Clear definition of system boundary
- Use of abstraction and partitioning
- Automated tool support
  - e.g. CASE tools provide automated consistency checking

Disadvantages

- Little use of projection
  - even SRD's 'perspectives' are not really projection
- Confusion between modeling the problem and modeling the solution
  - most of these techniques arose as design techniques
- These approaches model the system, *but not its application domain*
- Timing & control issues are completely invisible
  - Although extensions such as Ward-Mellor attempt to address this

References

van Vliet, H. "Software Engineering: Principles and Practice (2nd Edition)" Wiley, 1999.

> In common with many authors, van Vliet does not separate structured analysis from structured design. This makes sense because the two are intended to be used together. Section 11.2.2 gives a nice overview of the whole process, based on Yourdon's notations (SASS & descendents). He also gives a good introduction to SADT in section Section 9.3.3.

Svoboda, C. P. "Structured Analysis". In Thayer, R. H and Dorfman, M. (eds.) "Software Requirements Engineering, Second Edition". IEEE Computer Society Press, 1997, p255-274

> Excellent overview of the history of structured analysis, and a comparison of the variants

Davis, A. M. "Software Requirements: Analysis and Specification". Prentice-Hall, 1990.

> This is probably the best textbook around on requirements analysis, although is a little dated now.