SEIS 610

Agenda

- Review
- Classical Analysis
  - Specification Document (12.1)
  - Information Specifications (12.2)
  - Structured Systems Analysis (12.3)
  - Semi Formal Techniques (12.5)
  - Finite State Machines (12.7)
  - Petri Nets, Other Formal Methods (12.8, 12.9)
  - Comparison (12.11)

Review Requirements – Business Model (11.1-11.5)

- First: Understand the domain
  - Acquire familiarity with application domain (Important)
  - You can't hope to automate a process for somebody without understanding the problem they are trying to solve.
  - This is why we have a glossary!
- Business Model
  - Business model of the 'domain'
  - How do they make money?
  - Why is this product valuable

Review: Requirements – Business Model

- Why should/should software help?
  - How much will the software cost to create?
  - Is the software going to be sold or used internally?
  - If it is going to be sold as a product:
    - For how much?
    - How much do competitive products cost?
    - What features do competitive products have?
    - Where will this product fit within the product offerings?
    - Can you think of two people besides family members who will buy it??

Review: Business Model

- Initial Requirements
  - User Stories
    - **As a user, I would like to _____ so that I can ______**
  - Use cases
    - More formal, user/system interaction
- Functional Requirements
  - An action the target must be able to perform
  - Created during requirements and refined during analysis workflows
- Non-functional Requirements
  - Specifics related to the product itself
    - Platform constraints, response times, reliability
  - Best addressed during requirements and analysis, but may have to be handled during design.

Chapter 12

- Classical Analysis

Classical Analysis (12)

- Specification Document (12.1)
  - Documents should give engineers/developers and client/stakeholders a good understanding of what the product **must do**.
  - No matter what methodology is used
- This must contain acceptance criteria
- Solution strategies and plans are proposed.
- Bottom line
  1. Document gives good feeling about the product
  2. Document gives good feeling about acceptance criteria
  3. Document gives good feeling about a select solution strategy. (in class world)

Constraints

- Read page 360 & 361
- Where do constraints go in the Rational Unified Process?

Solution Strategy

- Where does the solution strategy go in the Rational Unified Process?

Informal Specifications (12.2)

- Written in a natural language such as English
- Page after page of text describing what the product should do.
- Weakness is ambiguity
  - Lawyers battle this constantly!
  - So do teachers
  - Most of us are not lawyers
  - No matter how well we explain in English we are always missing something.
- Realistically, correctness proof's such as 12.2.1 are not going to happen.

Informal Specifications

- What are these in the rational unified process?
- How do we battle ambiguity?

Glimpse: Analysis in OO/RUP

- Specification Document
  - Requirements document
    - Use cases, stories, FURPS, etc.
  - Design document
    - Sequence diagrams
    - Class diagrams
    - state, activity diagrams
  - Test Document
    - Detailed with traceability
- Informal Specification issues are mitigated with the 3-leg stool!
  - What you want, How we are gong to it, What it really does!

Structured Analysis (12.3)

- Data Flow Diagram (DFD)
- A graphical representation depicting the flow of data in an 'information' system.
- A tool analysis's can use to collect information necessary for system requirements.

Data Flow Diagrams

- Good sources
- ~~http://web.simmons.edu/~benoit/lis486/s13/readings/Notes-Analysis-2~~
- https://www.visual-paradigm.com/tutorials/data-flow-diagram-dfd.jsp

Data Flow Diagram

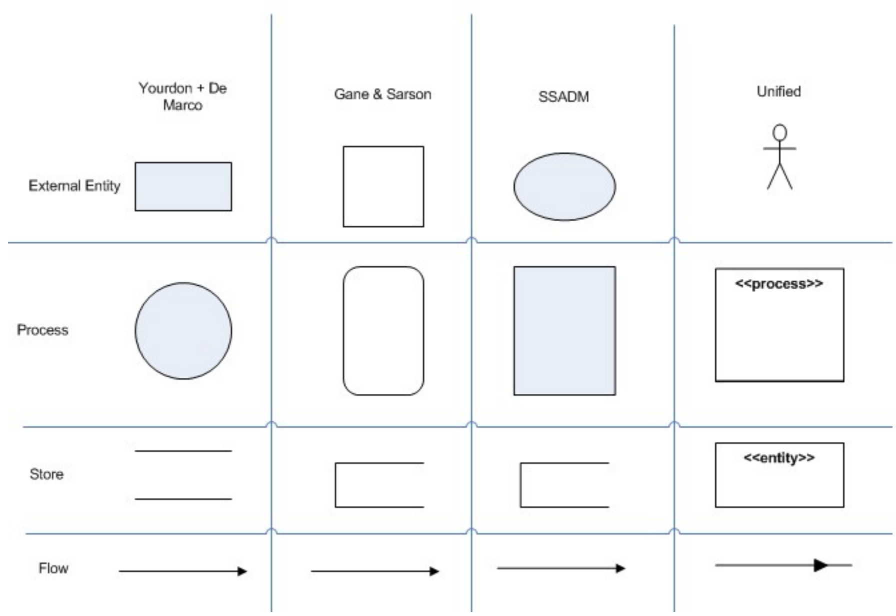
- Identify Data Flows
  - Requirements
    - Use cases/etc.
  - Rapid Prototype
- Data flows
  - Start with source or destination of data
  - OR Start with data store
- Data
  - Transformed by processes

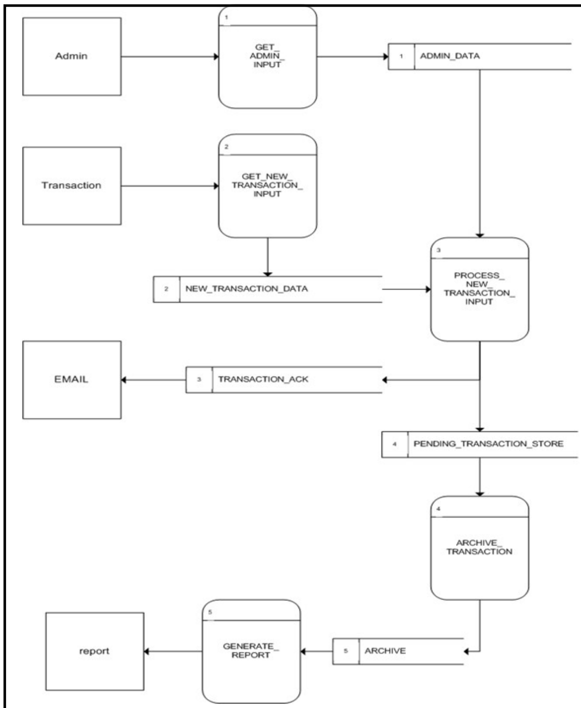
Structured Systems Analysis (Data Flow Diagrams) (12.3)

- Process oriented
- Help identify existing business processes
  - Hint: Think understanding the domain
- Can help re-engineer business processes
- Examines how data flows through the system

Data Flow Diagrams (comprised of)

- External Entity (source or destination-'sink' of data)
  - Human
  - System
  - Subsystem
  - External to the system we are studying
- Process
  - Business activity where manipulation and transformation of data occurs
- Data Store
  - Represents persistent storage





http://www.paulherber.co.uk/visio-dfd/examples.php

Data Flow Diagrams

- A process must have inputs
- A process must have outputs
- Source can have outputs
- Sink can have inputs
- Source/Sink <u>can have inputs and outputs</u>
- Data store <u>can have inputs and outputs</u>

Sources and Sinks (Source or Destination)

- Sources & sinks are referred to as external entities because they go outside the system.
- We don't need to consider the following:
  - Interactions that occur between sources and sinks
  - What a source or sink does with the data or how it operates (a black box)
  - How a control or redesign a source or sink since the info system deals with data as they are (what goes in and out of the box)
  - How to provide sources and sinks direct access to stored data because they cannot manipulate the data: the system must receive or distribute data between the system and its environment

Processes

- **No process can only have outputs**. That would be making data from nothing. If it only has outputs, it must be a source.
- **No process can only have inputs**. If it has only inputs, then it must be a sink.
- A process should have a verb in its name.

Data Store

- Data **can't** move from one store to another store, it must be moved by a process.
- Data cannot move directly from a source to a data store
- Data must be moved from a data store to a sink

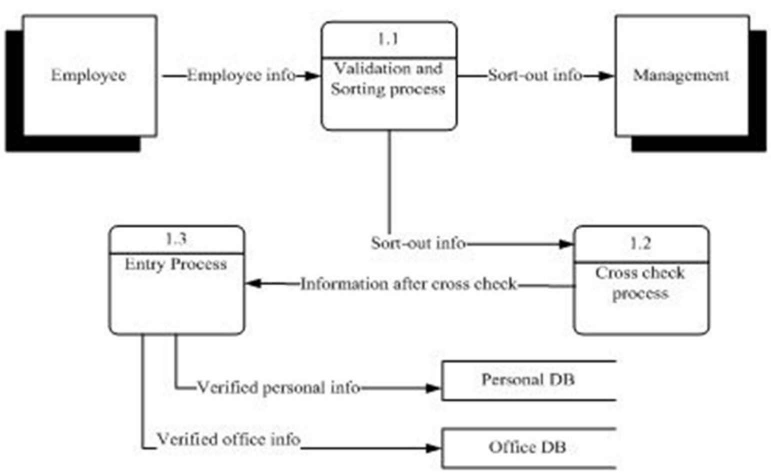
Practice Exercise

- Payroll DFD Diagram

How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

Sample Data Flow



How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which ‘processes’ to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

How to perform structured systems analysis

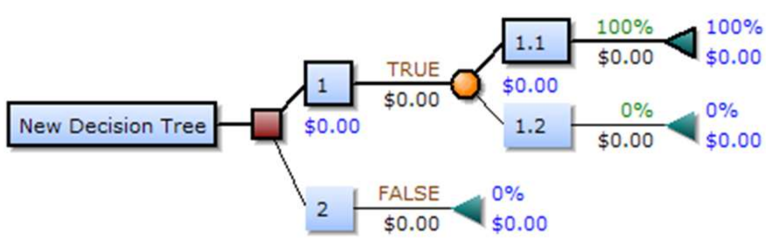
1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

How to perform structured systems analysis

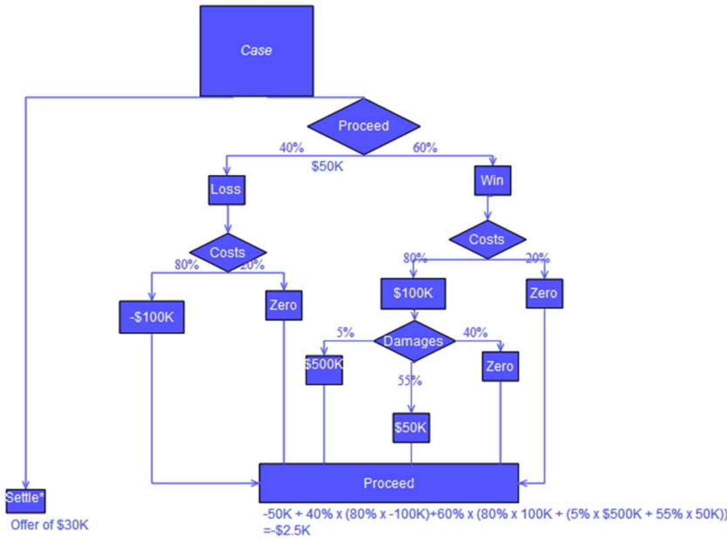
1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes (eek! How do I do that!?!!)
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

Define the logic of the processes

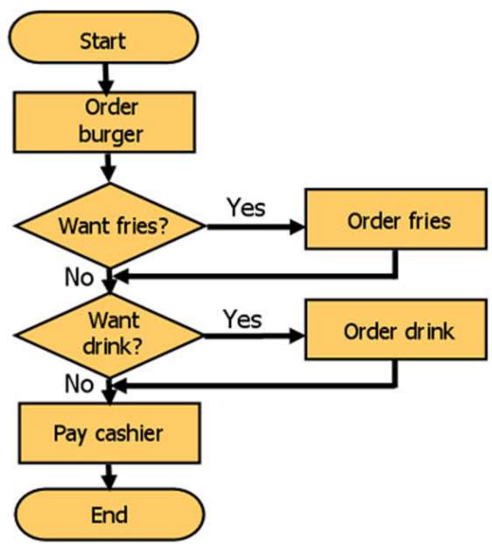
- Decision Tree



Decision Tree using flowchart symbols



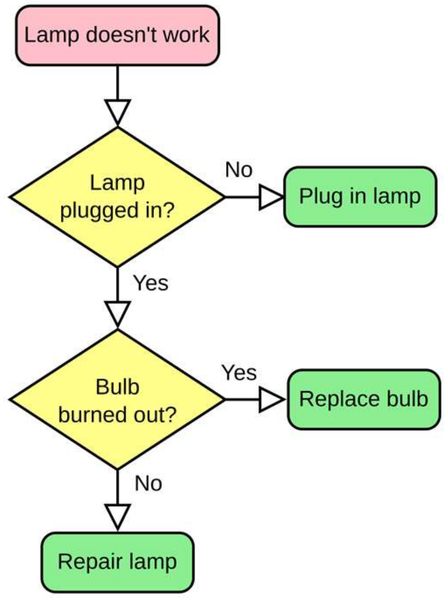
Activity Diagrams



- “Old fashioned flowcharts”
- Use for methods with difficult logic.

- http://www.teach-ict.com/as_a2_ict_new/ocr/A2_G063/331_systems_cycle/analysis_tools/miniweb/images/flowchart.jpg

Even cleaner flowchart



How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

How to perform structured systems analysis

1. Draw the dataflow diagram
2. Decide which sections to computerize
3. Determine the details of the data flows
4. Define the logic of the processes
5. Define the data stores
6. Define the physical resources
7. Determine IO specifications
8. Perform sizing
9. Determine the HW requirements

Other semiformal techniques.

- PSL/PSA
- SADT
- SREM
- Structural Analysis
- Important that you know the existence of these
- Why???

Skip 12.6, Entity Relationship

- Skip 12.6

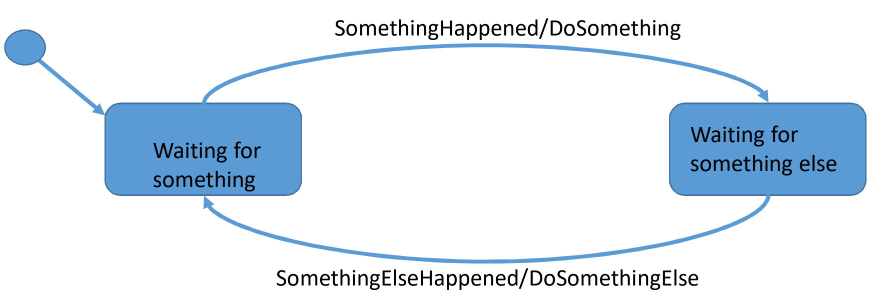
12.7 Finite State Machines

- States represented by rectangle or circle

- Events Represented by arrows

- Event name and action performed are on top of arrow.
  - Sometimes action is not shown.

State Machine Basics



States:
1. Waiting for Something
2. Waiting for Something Else

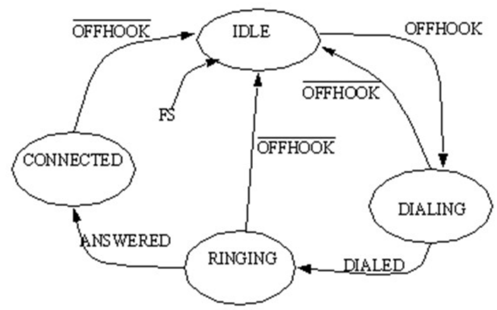
Events:
1. Something Happened
2. SomethingElseHappened

Actions
1. DoSomething
2. DoSomethingElse

1. We start in Waiting for Something.
2. S

Finite State Machines

- State Diagrams
- One way of representing interesting state logic.



http://www.thelearningpit.com/hj/plcs_files/plcs-353.gif

Finite State Machines

Toaster Example

Petri Nets

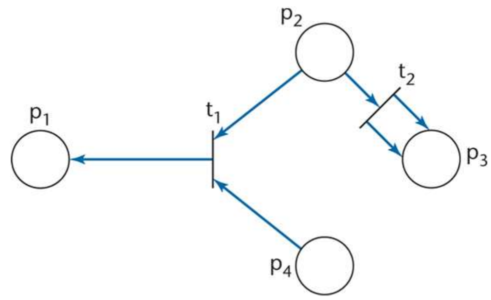
- Closely related to state machines
- Conceptually can deal with timing issues
  - Note, missing in state diagrams …BUT
  - Timeouts are events that can be scheduled to deal with timing issues

Petri Nets

- 4 Parts
  - Set of places
    - Circles
  - Set of transitions
    - Solid bar
  - Input function
    - Arrows from places
  - Output Function
    - Arrows to places

Petri Nets (contd)

- Set of places P is $\{p_1, p_2, p_3, p_4\}$
- Set of transitions T is $\{t_1, t_2\}$
- Input functions:
  $I(t_1) = \{p_2, p_4\}$
  $I(t_2) = \{p_2\}$
- Output functions:
  $O(t_1) = \{p_1\}$
  $O(t_2) = \{p_3, p_3\}$



Petri Nets (contd)

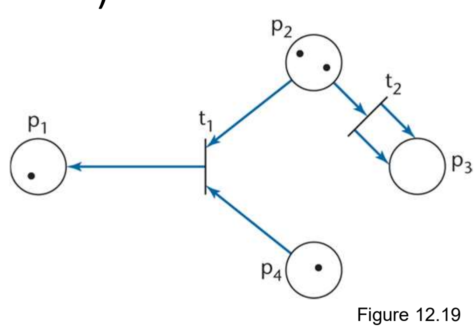


Figure 12.19

- Four tokens: one in p_1, two in p_2, none in p_3, and one in p_4
  - Represented by the vector $(1, 2, 0, 1)$
- A transition is enabled if each of its input places has as many tokens in it as there are arcs from the place to that transition

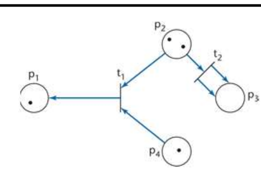
Petri Nets (contd)

- Transition t_1 is enabled (ready to fire)
  - If t_1 fires, one token is removed from p_2 and one from p_4, and one new token is placed in p_1

- Transition t_2 is also enabled

- Important:
  - The number of tokens **is not conserved**

Petri Nets (contd)



- Petri nets are indeterminate
  - Suppose t_1 fires

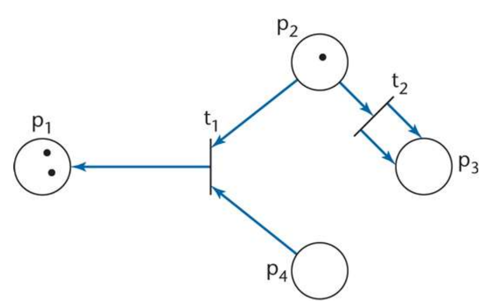


Figure 12.20

- The resulting marking is (2, 1, 0 ,0)

Petri Nets (contd)

- Now only t_2 is enabled
  - It fires

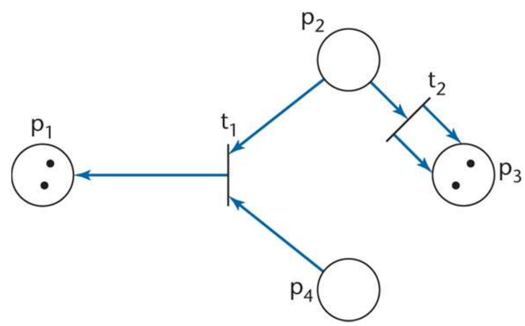


Figure 12.21

- The marking is now (2, 0, 2, 0)

Petri Nets

- Example

Other formal methods

- Z
  - Formal specification language
  - Next conference is July 2014, buy your tickets now!
- Anna
- Gist
- CSP
- VDM
- Once again, why should we know these exist?

Comparison of Specification Methods

- Very few of the exotic methods are in widespread use
- Analysis (and design) are problems software engineers continue to grapple with.
- If the method you are using is coming up short, it may be time to walk through engineering history
- Perhaps your problem has already been solved!

Comparison of Specification Methods

- Why study classical analysis, when object-oriented analysis is considered so much better?

Why study classical analysis, when object-oriented analysis is considered so much better?

- Understanding the problem domain.
- Understanding the business model.

- The end!