A Process for Combining Object Oriented and Structured Analysis and Design

Dale M. Rickman

Raytheon Systems Company

8/9/2000

dmrickman@raytheon.com

Why Combine Methodologies?

- Object Oriented (OO) Methodologies have some Advantages over Structured Methods, but do not Contain all of the Tools and Techniques Needed to Support a Large Information System Design
- Standard Structured Analysis Methods do not Support Reuse as Well as Object Oriented Methods
- By Combining the Methods in one Process we can Take Advantage of the Strengths of Both Methodologies!

Advantages and Disadvantages of Object Oriented Methodologies

- Advantages
  - Supports Inclusion of COTS and Reusable Components
  - Supports Development of Reusable Components
  - Provides Insight into Data Relationships
- Disadvantages
  - Lack of a System Functional Model can lead to Missed Requirements
  - Difficult to Model System Performance and Sizing
  - Little Support for Identifying the Objects for an Optimal Design
    - Difficult to “See” Interface Complexity

Advantages and Disadvantages of Structured Methods

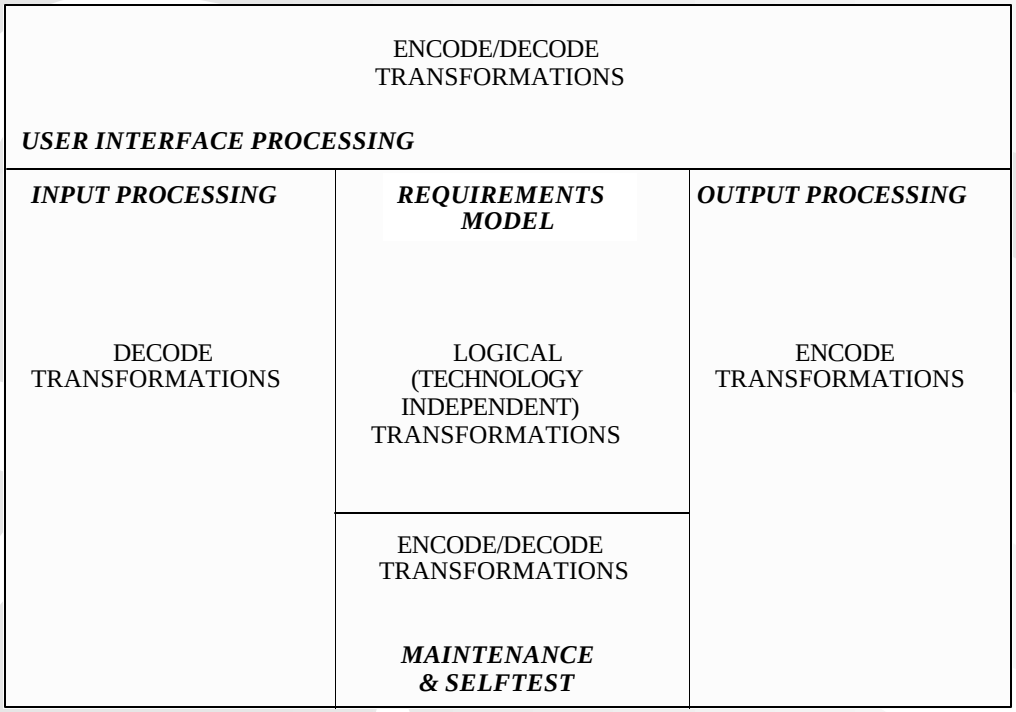
- Advantages
  - Mature Discipline for Large Systems Design
  - Better Understood by Developers and Customers
  - Contract Requirements are Specified in Terms of Functions
  - Provides Complete (Functional) Requirements Model
- Disadvantages
  - **Does Not Readily Support COTS/Reuse**
    - Requirements Developed Top-Down Do Not Map Well to Reusable Components

Hatley-Pirbhai (H-P) Methodology

- Addresses the Short Fall of Standard Structured Methods
  - Supports Inclusion of Reusable Components
  - Enhanced Requirements Model Addresses “Glue” Code
- Provides 3 Requirements Models
  - Process Model - Models Functional Requirements
  - Control Model - Models System Control
  - Information Model - Models Data Relationships
- Provides 2 Architecture Models
  - Architecture Flow Model - Models Data Flows Between Components
  - Architecture Interconnect - Models Physical Connection of Components
- All Models are Tied Together via a Data Dictionary

Hatley-Pirbhai Enhanced Requirements Model

- Enhancements include Derived Requirements Due to:
  - Non-perfect World
    - Security Requirements
    - Availability Reqs
  - Design Decisions
    - Environment
    - Distributing Components
    - Using COTS



Combining OO into Hatley-Pirbhai

- Combining Object Oriented Methods and Hatley-Pirbhai Provides an Excellent Toolset for Designing Systems
- Object Oriented Models Used as:
  - Replacement for Hatley-Pirbhai Information Model
  - Architecture Model for Software Module Design, if Appropriate

The Process of Combining OO and H-P

- Analysis and Design Steps:
  - Build Context Diagram
  - Use Functional Decomposition to Partition Requirements
  - Build an OO Model to Understand Data Relationships
  - Build a Control Model to Understand System Control
  - Enhance Requirements Model for Non-perfect World and Design Decisions
  - Group Requirements and Allocate all Requirements, Data Stores and Control to Architecture Components
  - Build Architecture Flow and Interconnect Diagrams
  - **Iterate** until Models are Consistent and Design Optimized
    - Populate the Dictionary as you go

Implementing the Process

- The Order of Steps is not Important
  - Can Start with any Requirements Model or even Architecture
  - The Key is to Ensure all Models Integrate at Some Point
- When 1st Level Architecture is Complete, Repeat Process for Each 1st Level Architecture Component (e.g., Subsystem)
  - Each Subsystem is Treated as a New System to be Architected
  - Can Now Partition the Problem and Design in Parallel
- When Designing at Software Configuration Item Level, Choose Appropriate Design Method (OO or Structured)
- Use the Object Oriented Information Model as the Architecture Model for Software being Developed Using OO

OO Designs are Fundamentally Different from Stuctured Designs

- Stuctured Design
  - Control Modeled Separately from Functions
  - Single “Smart” Controller
  - Analogous to a Military Model
  - Very Efficient
- OO Design
  - Control Partitioned into Each Object
  - Each Object is Stand-alone
  - Service Oriented Model
  - Very Flexible & Reusable

Conclusion

- Systems Engineering Has Changed Over the Last 5 Years
  - Minimal New Large Systems Development
  - Better, Faster, Cheaper Drives Us to Using More COTS and Reusable Components
- New Procurement Focus Requires New Processes
  - Need to Expand the “Toolbox”
  - Use New Methodologies, Like Object Oriented
- Combining Methodologies Provides Better Insight into the Problem and Generates Better Solutions.