Introduction to Pattern Oriented Analysis and Design (POAD)

Instructor: Dr. Hany H. Ammar
Dept. of Computer Science and Electrical Engineering, WVU
Outline

- Review of Design Patterns
  - The Lifecycle of a Pattern
  - Examples of Design Patterns
    - The Command Pattern
    - The Observer Pattern
    - The Strategy Pattern
- Pattern Oriented Development
  - The Analysis phase
  - The Design phase
  - The design refinement phase
- The Feedback Control Example
The Lifecycle of Patterns

(From the Reference “Pattern-Oriented Analysis and Design”, Sherif M Yacoub and Hany H. Ammar Addison-Wesley Inc., 2004)

Legend

<table>
<thead>
<tr>
<th>Activity</th>
<th>Product</th>
</tr>
</thead>
</table>

Phase I: MINING
Author World

Real World Projects

Discover Patterns

Pattern Mining

Incident Occurrence of a Pattern

Document

Analyze/Rule of Three

Preliminary Documentation

Author Version

Modification

Feedback

Phase II: POLISHING
Community Pattern World

Pattern Polishing

Reusable Version

Pattern Reuse

Phase III: REUSE
Pattern User World
Review of Design Patterns

Examples of Design Patterns (From the Design Patterns CD by Gamma et al, Addison-Wesley Inc., 1998)

The Command Pattern: Encapsulate a request as an object

![Diagram of the Command Pattern]

- Application
  - Add(Document)

- Menu
  - Add(Menu Item)

- MenuItem
  - Clicked()
  - command

- Command
  - Execute()

- Document
  - Open()
  - Close()
  - Cut()
  - Copy()
  - Paste()

- command → Execute()
Review of Design Patterns

The Command Pattern

From the Design Patterns CD by Gamma et al, Addison-Wesley Inc., 1998
Review of Design Patterns

Examples of Design Patterns

The Observer Pattern: when one object changes state, all its dependents are notified and updated automatically.

Model View Controller example
Examples of Design Patterns

The Strategy Pattern: lets the algorithm vary independently from clients that use it
Pattern Oriented Development

- Design Patterns are used in an ad-hoc strategy for design refinement.
- They are also used to address a set of design problems without any guidance of how these patterns can be glued or interface together.
- Is there a way to use design patterns as building blocks or as components in the design of systems?
Pattern Oriented Development

Pattern Oriented Analysis and Design (POAD)

- The process aspects of POAD explains the phases and steps to develop an application design using patterns
- The POAD process has three phases:
  - Analysis
  - Design
  - Design Refinement
Pattern Oriented Development

- Analysis
  - Design
    - Design Refinement
      - a logical model is developed and patterns are selected
        - where patterns are glued together to produce a detailed pattern-level diagram
          - An initial class diagram, and a more dense and profound class diagram, and sequence diagrams are developed
The POAD process

(a) Overall POAD

- Analysis
  - Design
    - Design Refinement

(b) Analysis

- Pattern Library
  - Acquaintance
    - Retrieval
      - Candidate Patterns
        - Selected Patterns

(c) Design

- Concretization
  - Specialization
    - Instantiating Pattern Internals
      - Domain Specific Detailed Pattern-Level Diagrams
        - Develop Class Diagrams
          - Initial UML class diagram

(d) Design Refinement

- Optimization
  - Reduction
    - Merging & Grouping
  - Detailed Pattern-Level Diagrams

- Design Optimization
  - Optimized class diagram
POAD Analysis Phase

- Develop use case diagrams to identify the problems to be solved and the possible breakdown of the application as a set of logical components.
- Acquaintance with relevant pattern databases to get the analyst familiar with existing solutions.
- Retrieval of patterns from the domain specific databases to select a set of candidate patterns in an automated fashion.
- Selection of patterns from a set of candidate patterns for possible inclusion in the design process.
Construct Pattern-Level Models

- Create an instance for each selected pattern by describing the patterns and their constituents in an application specific context
- Define how these instances are related to each other
- The semantic of a dependency relationship used between patterns has a "uses" meaning
The Feedback Control Example

From Pattern-Oriented Analysis and Design, Sherif M Yacoub and Hany H. Ammar Addison-Wesley Inc., 2004

- Block diagram for a feedback control system

Reference Input

Error (Actuating) Signal

Feed forward Elements

Controlled Output

Plant.

Feedback Data

Feedback Elements

Measurement
The Feedback Control Example

**POAD Analysis Phase (Pattern Selection)**
- The *feedforward* component implements some sort of a control strategy (instance of Strategy pattern)
- The *feedback* component receives measurements and applies a feedback control strategy
- In the *error calculation* component, the feedback controller notifies the error calculation unit with the feedback data (instances of the Observer pattern)
- Data of different types need to be exchanged between the framework components (Measurement, Feedback Data, input data, and error data)
The Feedback Control Example

This is an architectural pattern based on the data flow architectural style.
The Feedback Control Example

Pattern-Level with Interfaces

Context

Update

setData

getDataType

<<Strategy>> FeedforwardStrategy (from POAD1-Feedback)

<<Observer>> FeedbackObserver (from POAD1-Feedback)

<<Observer>> ErrorObserver (from POAD1-Feedback)

<<Strategy>> FeedbackStrategy (from POAD1-Feedback)

<<Blackboard>> Blackboard (from POAD1-Feedback)
The Feedback Control Example

Detailed Pattern-Level diagram
The Feedback Control Example

Instantiating the ErrorObserver pattern

```
<<Observer>>
ErrorObserver
(observerState)
Update()
FeedbackSubject
(subjectState)
GetState()
AbstractSubject
Attach()
Detach()
Notify()
AbstractObserver
Update()
```

Update

Notify
The Feedback Control Example

Instantiating the *FeedforwardStrategy* pattern
The Feedback Control Example

Instantiating the *Blackboard* pattern

```
setData()
getData()
```

```
DataHolder
getData
setData
```

```
ErrorData
MeasuredData
FeedbackData
```

```
<<Blackboard>>
Blackboard
(from POAD1-Feedback)
```
The Feedback Control Example

The initial class diagram
The Feedback Control Example

The refined class diagram ready for code generation.
The Feedback Control Example

Object Collaboration Diagram

- F_controller : AbstractController
- Strategy1 : ControlStrategyA
- FB_Strategy : FBControlStrategyA
- Measurement : MeasurementSubject
- Error : ErrorObserver
- Feedback : FeedbackSubjectObserver
- TheBlackboard : Blackboard

Activities:
1. Apply (DataHolder*)
2. Notify()
3. Update()
4. GetState()
5. MeasurePlant()
6. Update()
7. FBApply()
8. Update()
9. Notify()
10. Update()
11. Getstate()
12. GetInput()
13. Analyze()
14. Update()
15. Control (DataHolder*)