Application Facades:
Bringing Usability to Business Objects

Presented by:

William F. Nazzaro

E-Mail: bill@williamnazzaro.com
Object Technology’s State of the Union

• Problem
  – Many designers
    » Unknowingly make their business objects less reusable by including application specific rules and processing inside their objects
    » Are solving the immediate problem without taking a “quick” step back and looking at a potentially more flexible solution

• Solution
  – Utilize an application facade
Agenda

- Application Facades - Introduction
- Three-Tier Architecture
- Application Facades - In Depth
- Examples
- Other Uses
**Application Facades**

**Introduction**

- **What are they?**
  - *A logical grouping of business objects for the purpose of providing a simplified interface*
    - *Often*
      - `Specific to this application`
      - `Represent something in the real world`
    - `Normalization tends to eliminate them`
    - `Logical view of business objects`
    - `Similar in concept to a view in a database`
    - `Completely derivable`

- **How are they used?**
  - *Tend to glue interface objects to the business objects*

*I would like to acknowledge Martin Fowler for his work in this area*
Application Facades

*When Should I Focus on Them?*

- It’s more appropriate to focus on them during design

- Why didn’t I focus on them during analysis?
  - Analyst focuses on discovering the “essential” objects
  - Typically, we don’t focus on application facades during analysis because there are so many other issues to be concerned with
  - Now we verify the existence of objects to support the use cases and establish a checkpoint to ensure their discovery and integration

- What if I did find application facades during analysis?
  - Sometimes we do discover application objects during analysis
  - Sometimes the customer won’t agree on a class diagram or some other deliverable until they see them

- Where can I look to find these types of objects?
  - Use cases and screen sketches/prototypes are good sources
Three-Tier Architecture

• Interface Objects
  – *Layer contains objects like*
    » *Screens*
    » *Menus*

• Business Objects
  – *Application specific*
    » *Contains the local business requirements of the application being developed*
  – *Corporate wide*
    » *Contains objects that are relevant to the whole organization*

• Storage Objects
  – *Encapsulate the storage of data to a relational or object database*
Interface Objects

• Take data held in the business objects and display it on the screen in a particular format (e.g., text, graphics, etc.)

• Aware of their dependency on business objects

• Have little to do with the business object’s semantics
  – *Focus on the presentation of a business object*
  – *Maps the formatted information onto its business object*

• Do **not** “understand” the attributes

• Do **not** act upon the attributes, except to display them
Business Objects

• Represent objects in the problem domain
  – *Contains the business rules that govern the object*

• Know about each other or their storage objects

• Should contain
  – *Attributes*
  – *Operations*
  – *Rules of the business objects they represent*

• Should **not** include any
  – *Presentation*
  – *Interface control mechanisms*
### Application Facades

**In Depth**

- **Where does the facade exist?**
  - *Between the interface layer and the business layer*

- **Application facades**
  - *Provide a series of custom facades for the interface objects*
  - *Select information from the various business objects and organize that information for the appropriate interface objects*

- **Application facade’s structure**
  - *Attributes are derived from the business objects*
  - *Operations are identified for each attribute or group of attributes*
Application Facades

*Pros and Cons*

**Pros**

- Provides a certain amount of logical independence in the face of restructuring the business objects
- Allows the same business objects to be seen by different users in different ways (at the same time)
- Designer’s perception is simplified
- Provides a new layer of security for information that shouldn’t be accessed when using the business object a certain way

**Cons**

- Extra layer between interface and business
Application Facades

Attributes

• Derivable
  – Attributes that are located in a business object(s)

• Specific
  – Attributes that are specific to the application facade and are usually set through the context in which it’s being used

• What should I model?
  – If I’m early in design I just model the specific attributes
  – As I move to implementation I then model the derivable attributes
Application Facades

Operations

- **Public Operations**
  - *Gets* ~ Defines how information is retrieved from the business object(s) to fill the attributes
  - *Sets* ~ Defines how information is updated in the business object(s)
  - *Something that requires complex processing. Need to consider if it’s*
    - Local
    - Shared

- **Private Operations**
  - *Gets* ~ Defines how information is retrieved from the business object(s) to fill the attributes
  - *Sets* ~ Defines how information is updated in the business object(s)
  - *Default* ~ Indicates what values the attributes should be if we are creating a new business object(s)
  - *Validation* ~ Validation rules that are specific to this particular application
An Example

Transcript Sketch

Date: ________

Generic University Transcript

<table>
<thead>
<tr>
<th>Student ID</th>
<th>College ID</th>
<th>Advisor #</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>College Name</th>
<th>Advisor name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address 1</th>
<th>Major</th>
<th>Date admitted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Address 2</th>
<th>Date admitted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>City</th>
<th>Zip:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State</th>
<th>Zip:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Generic University Courses Taken

<table>
<thead>
<tr>
<th>Semester</th>
<th>College</th>
<th>Course #</th>
<th>Course name</th>
<th>Credit hours</th>
<th>Grade</th>
<th>Honor points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Generic University Degrees

<table>
<thead>
<tr>
<th>Degree</th>
<th>College</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Previous Degrees

<table>
<thead>
<tr>
<th>Degree</th>
<th>University</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


An Example
Initial Class Diagram

Person
- Address
- Name
- SSN
- Telephone

College
- Address
- CollegeID

Degree
- Date Conferred
- Name
- Type

Track
- Description
- Name

Student
- GPA
- Student ID
- Year

Instructor
- Department
- Employee #
- Qualifications

FullTimeInstructor
- Salary
- Vacation

AdjunctInstructor
- Wage

InstructorType

GU Degree
- Degree

Previous Degree
- Degree
- University

Course
- Course #
- Credit Hours
- Description
- Title

CourseOffering
- Seats
- Section
- Semester
- Time

CourseEnrollment
- Grade

Student enrolls in

Person advises

Instructor teaches

Course requires

Degree confers

Track requires

Person controls

Student registers for

Course fills a seat in

Previous Degree confers

CourseOffering offers

Degree confers

Instructor teaches

Person advises

Student enrolls in

Course requires

Degree confers

Track requires

Instructor teaches

Person advises

Student enrolls in

Course requires

Degree confers

Track requires

Instructor teaches

Person advises

Student enrolls in

Course requires

Degree confers

Track requires

Instructor teaches

Person advises

Student enrolls in

Course requires

Degree confers

Track requires

Instructor teaches

Person advises

Student enrolls in

Course requires

Degree confers

Track requires
An Example

Detailed Class Diagram - 1

Application Facade

Student
- ActualCompletionDate
- AdmissionDate
- CreditHours
- Dependents
- DismissalDate
- EnrolledProgram
- GPA
- LastRegistrationDate
- Probation
- ReasonForDismissal
- ScheduledCompletionDate
- StudentState
- Student ID
- TransferCredits
- Year
- AcademicAdvisor: FullTimeInstructor
- Enrollments: List <CourseEnrollment>

StudentTranscript
- DateGenerated
- Dependents
- Student: Student
- Type [Official, Unofficial]

+ AddDependent
+ CreateTranscript
- Change
- GetCourseInfo
- GetGUDegrees
- GetPrevDegrees
- GetStudentInfo
+ PrintTranscript
+ Update

1 <- has 1
Other Uses

- Another use would be to provide a facade between the business object(s) and their storage object(s).
- The business object(s) would then be shielded from the underlying structure of the database.
- This facade is then responsible for managing the “impedance mismatch” between OO and RDBMS.
- Using the facade will allow greater flexibility in the future, but again with the cost of an additional layer.
Agenda

☑ Application Facades - Introduction
☑ Three-Tier Architecture
☑ Application Facades - In Depth
☑ Examples
☑ Other Uses
Application Facades:
Bringing Usability to Business Objects

Presented by:
William F. Nazzaro

E-Mail: bill@williamnazzaro.com