OBJECT-ORIENTED ANALYSIS Part II

Object-Oriented and Classical Software Engineering
Sixth Edition, WCB/McGraw-Hill, 2005
Stephen R. Schach
srs@vuse.vanderbilt.edu

Overview

- Extracting the control classes: The Osbert Oglesby case study
- Refining the use cases: The Osbert Oglesby case study
- Use-case realization: The Osbert Oglesby case study
- Incrementing the class diagram: The Osbert Oglesby case study
- The specification document in the Unified Process
- More on actors and use cases
- CASE tools for the object-oriented analysis workflow
- Challenges of the object-oriented analysis workflow

Extracting the Control Classes: The Osbert Oglesby Case Study

- It is also usually easy to extract control classes
  - Each nontrivial computation is generally modeled by a control class
- In the case study there are four computations
  - Determining the maximum price that Osbert should offer for a
    - Masterpiece
    - Masterwork, or
    - Other painting
  - Determining if there is a new trend in art purchases

- There are therefore four initial control classes

Compute Masterpiece Price Class
Compute Masterwork Price Class
Compute Other Painting Price Class
Compute Future Trends Class

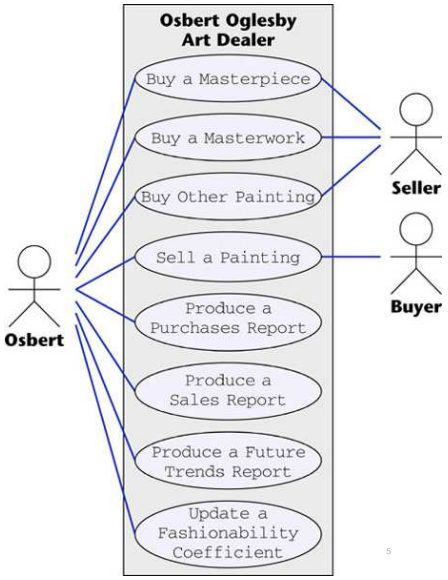
Refining the Use Cases: The Osbert Oglesby Case Study

- The pricing algorithm treats the three types of paintings differently
- Use case `Buy a Painting` must therefore be refined into three separate use cases
  - `Buy a Masterpiece`
  - `Buy a Masterwork`
  - `Buy Other Painting`

- Use case `Produce a Report` also needs to be refined
  - The purchases report and the sales report use simple data extraction — the future trends report involves computation
  - All three reports use their own boundary classes
- For both these reasons, the `Produce a Report` use case must be refined into three use cases
  - `Produce a Purchases Report`
  - `Produce a Sales Report`
  - `Produce a Future Trends Report`

Third Iteration of the Use-Case Diagram

- Implications for the remaining UML diagrams include:
  - The description of the new `Buy a Painting` use case (see over) must be split into three separate descriptions
  - The description of the `Produce a Report` use case must be split into three separate descriptions



Use Case `Buy a Masterpiece`

Brief Description

The `Buy a Masterpiece` use case enables Osbert Oglesby to buy a masterpiece.

Step-by-Step Description

1. Osbert inputs the details of the masterpiece he is considering buying. These are
   - First name of artist
   - Last name of artist
   - Title of work
   - Date of work
   - Height
   - Width
   - Medium (oil, watercolor, other medium)
   - Subject (portrait, still life, landscape, other subject)

2. The software product responds with the maximum purchase price Osbert should offer. In more detail, the software product computes the coefficient of similarity between each painting for which there is an auction record and the painting under consideration for purchase. Question marks in the first or last name of the artist or in the title or date of the work are to be ignored.
   - The software product scores 1 for a match on medium, otherwise 0.
   - The software product scores 1 for a match on subject, otherwise 0.
   - It adds these two numbers, multiplies by the area of the smaller of the two paintings, and divides by the area of the larger of the two.
   - The resulting number is the coefficient of similarity.

   The software product finds the auctioned painting with the largest nonzero coefficient of similarity. If there is no such painting, Osbert will not buy the painting under consideration.

   The software product computes the maximum purchase price by adding to the auction price of the most similar work 8.5%, compounded annually, for each year since that auction.

3. If the seller accepts Osbert's offer, Osbert enters further details. These are
   - Date of purchase
   - Name of seller
   - Address of seller
   - Maximum purchase price determined by the algorithm
   - Actual purchase price
   - Target selling price

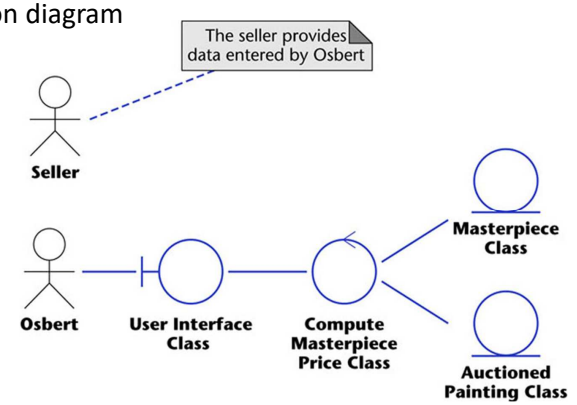
The description of class extraction is complete

Use-Case Realization: The Osbert Oglesby Case Study

- The process of extending and refining use cases is called *use-case realization*
- The verb "realize" is used at least 3 different ways:
  - Understand ("Harvey slowly began to realize that he was in the wrong classroom");
  - Receive ("Ingrid will realize a profit of $45,000 on the stock transaction"); and
  - Accomplish ("Janet hopes to realize her dream of starting a computer company") ทำให้บรรลุเป้าหมาย
- In the phrase "realize a use case," the word "realize" is used in this last sense

Use-Case Realization (contd)

- The realization of a specific scenario of a use case is depicted using an interaction diagram
  - Either a sequence diagram or collaboration diagram
- Consider use case `Buy a Masterpiece`
- We have previously seen
  - The use case
  - The description of the use case



The Four Classes That Enter into This Use Case

- **User Interface Class**
  - This class models the user interface
- **Compute Masterpiece Price Class**
  - This class models the computation of the price Osbert should offer
- **Masterpiece Class**
  - The computation involves comparing the masterpiece being considered with the masterpieces that have been previously auctioned
- **Auctioned Painting Class**
  - These masterpieces are all instances of **Auctioned Painting Class**

`Buy a Masterpiece` Use Case (contd)

- The Seller does not interact directly with the software product
  - Instead, the Seller provides data that Osbert enters into the software product
- This is indicated in the note (the rectangle with the top right-hand corner turned over)
  - There is a dashed line from the note to the item to which it refers, the Seller in this case

Osbert Oglesby wishes to buy a masterpiece.

1. Osbert enters the description of the painting.
2. The software product scans the auction records to find the price and year of the sale of the most similar work by the same artist.
3. The software product computes the maximum purchase price by adding 8.5%, compounded annually, for each year since the auction of the most similar work.

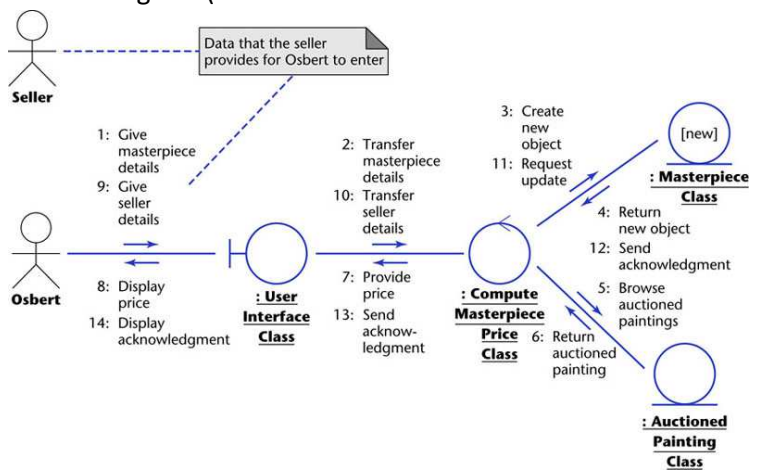
   Osbert makes an offer below the maximum purchase price—the offer is accepted by the seller.
4. Osbert enters sales information (name and address of seller, purchase price).

`Buy a Masterpiece` Use Case (contd)

- An executing software product uses objects, not classes
- Example:
  - A specific masterpiece is not represented by **Masterpiece Class** but rather by an object, a specific instance of **Masterpiece Class**
- Such an object is denoted in UML by **: Masterpiece Class**
- A class diagram shows the classes in the use case and their relationships
  - It does not show the objects nor the sequence of messages as they are sent from object to object
- Something more is needed

`Buy a Masterpiece` Use Case (contd)

- A collaboration diagram (of the realization of the scenario of the use case)



`Buy a Masterpiece` Use Case (contd)

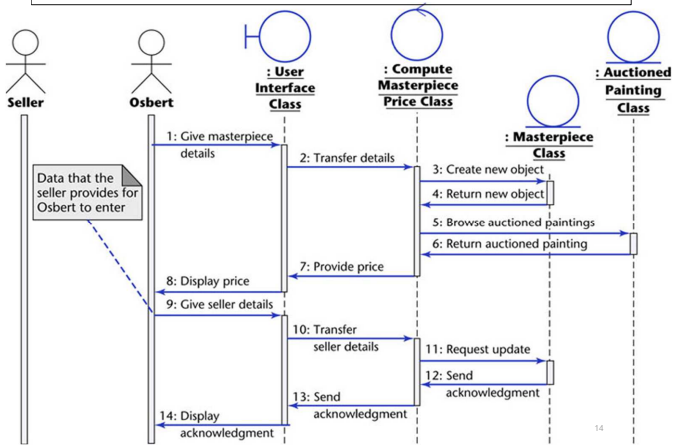
- Osbert will not approve the specification document unless he understands it
- Accordingly, a written description of the collaboration diagram is needed
  - The *flow of events*
- The flow of events of the collaboration diagram of the realization of the scenario of the use case

> Osbert inputs the details of the masterpiece he is considering buying (1). The software product then looks through all the masterpieces that have been auctioned to find the one closest to the masterpiece under consideration (2–6). It then computes the maximum price that Osbert should offer using the formula provided (7–8).
>
> Osbert now makes an offer. His offer is accepted, and he supplies details regarding the seller (9), which are then used to update the masterpiece data (10–14).

`Buy a Masterpiece` Use Case (contd)

- UML supports two different types of interaction diagram
  - *Collaboration diagram*
  - *Sequence diagram*
- Both contain exactly the same information, but displayed in different ways

Sequence diagram equivalent to the collaboration diagram (of the realization of the scenario of the use case)



`Buy a Masterpiece` Use Case (contd)

- The narrow rectangle on a lifeline (dashed vertical line) shows when the relevant object is active
- In the collaboration diagram, the [new] is inside the **: Masterpiece Class** object
  - In the sequence diagram, the object is shifted down so that its lifeline starts where the object is created
- The sequence diagram shows that every message of the scenario involves either
  - The instance of the user interface class **: User Interface Class** or
  - The instance of the control class **: Compute Masterpiece Price Class**

`Buy a Masterpiece` Use Case (contd)

- It also shows that every transfer of information from object A to object B is eventually followed by a transfer in the reverse direction
- These two facts are also true in the fully equivalent collaboration diagram, but are not as obvious in that format
- **Interaction Diagram**
- Software engineers can choose whether to use
  - A sequence diagram, or
  - A collaboration diagram, or
  - Both

  for each scenario

Interaction Diagrams

- The strength of a sequence diagram is that it **shows the flow of messages and their order unambiguously**
  - When transfer of information is the focus of attention, a sequence diagram is superior to a collaboration diagram
- A collaboration diagram is similar to a class diagram
  - When the developers are concentrating on the classes, a collaboration diagram is more useful than the equivalent sequence diagram

`Buy a Masterpiece` Use Case (contd)

- The seven previous figures depict different aspects of the use case `Buy a Masterpiece`
  - They use different notations and provide different levels of detail of the same activity
- Why do we construct so many related artifacts?
  - We examine this one activity from a variety of different perspectives to learn enough about it to ensure that the analysis workflow will be correct

`Buy a Masterwork` Use Case

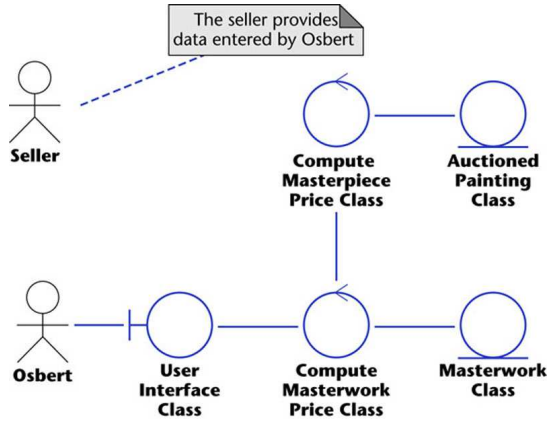
- The maximum price of a masterwork is computed by first treating the painting as if it were a masterpiece, and then adjusting the result

The Five Classes That Enter into This Use Case

- **User Interface Class**
- **Compute Masterwork Price Class**
  - This class models the computation of the price Osbert should offer
  - It creates a masterwork object and passes it to **Compute Masterpiece Price Class** as if it were a masterpiece
- **Compute Masterpiece Price Class**
- **Masterpiece Class**
- **Auctioned Painting Class**

`Buy a Masterwork` Use Case (contd)

- Class diagram (classes that enter into the use case)



`Buy a Masterwork` Use Case (contd)

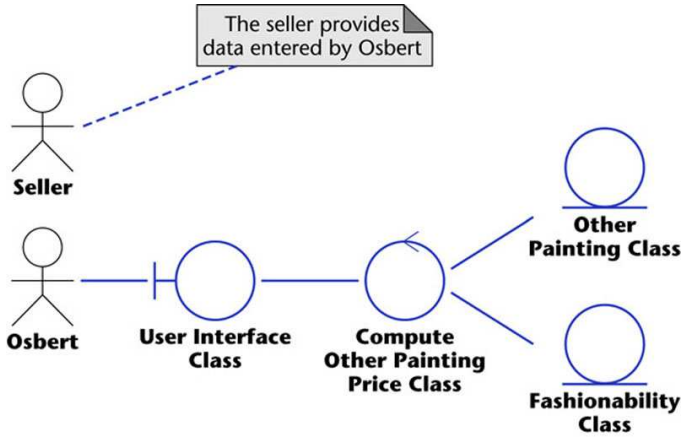
- One possible scenario of the use case

> Osbert Oglesby wishes to buy a masterwork painted in the seventeenth century.
> 1. Osbert enters the description of the painting.
> 2. The software product scans the auction records to find the price and year of the sale of the most similar work by the same artist.
> 3. The software product computes the maximum purchase price by adding 8.5%, compounded annually, for each year since the auction of the most similar work, and multiplying the result by (21 − 17)/(22 − 17), or 0.8.
>
> Osbert makes an offer below the maximum purchase price—the offer is accepted by the seller.
> 4. Osbert enters sales information (name and address of seller, purchase price).

- The remaining use cases are similar to those for the use case `Buy a Masterpiece`

`Buy Other Painting` Use Case

- Class diagram



- Scenarios and interaction diagrams (collaboration diagrams, sequence diagrams), and associated flows of events are left as an exercise

Modifying the Main Menu

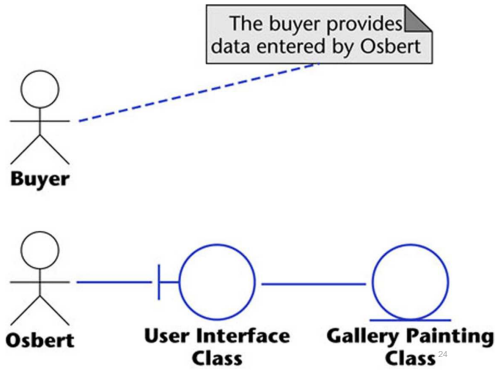
- The main menu must reflect buying the three different types of painting explicitly
- `Buy a Painting` must be replaced by
  - `Buy a Masterpiece`,
  - `Buy a Masterwork`, and
  - `Buy Other Painting`

- The revised screen is generated by **: User Interface Class**

The Remaining Five Use Cases

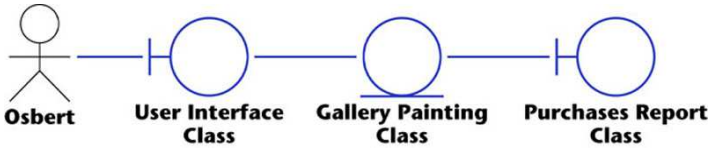
- Class diagrams are presented for the remaining five use cases
- The realizations are straightforward

`Sell a Painting` Use Case: Class Diagram

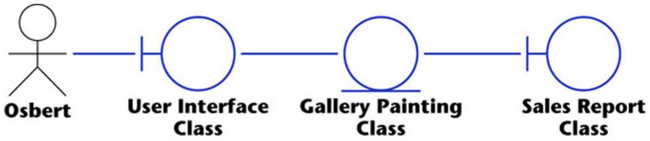


`Produce a Purchases Report` Use Case

- Class diagram

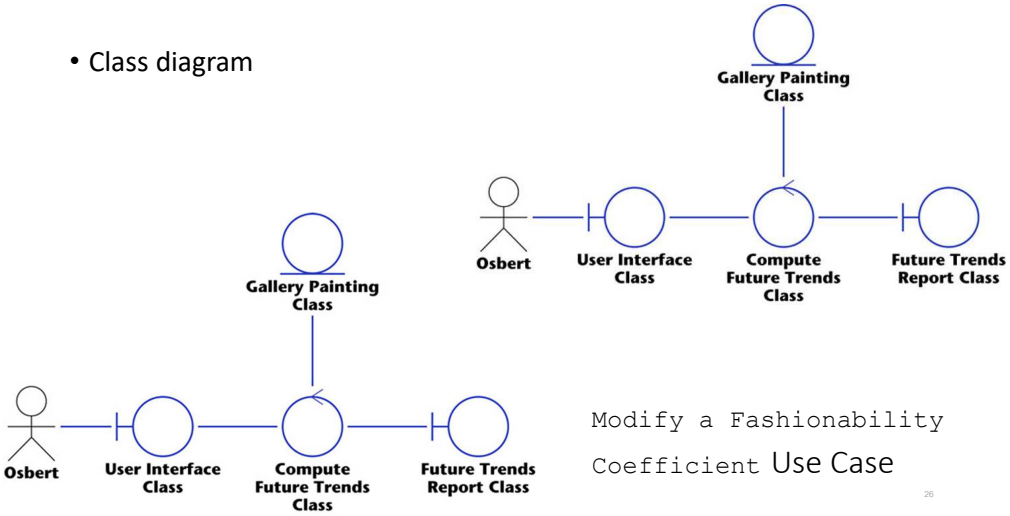


`Produce a Sales Report` Use Case



`Produce a Future Trends Report` Use Case

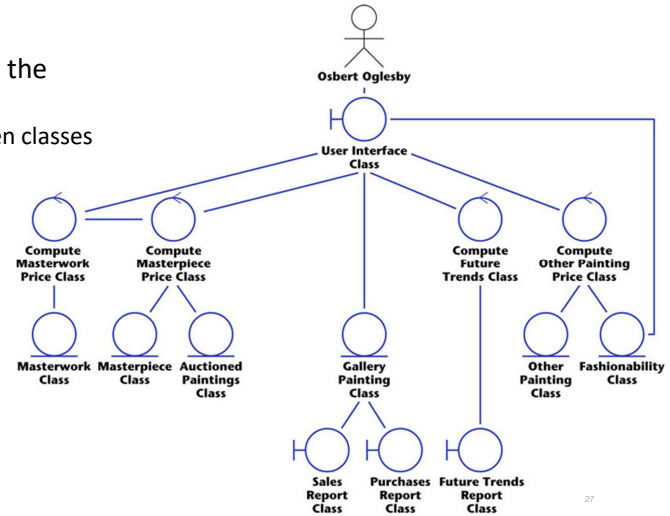
- Class diagram



`Modify a Fashionability Coefficient` Use Case

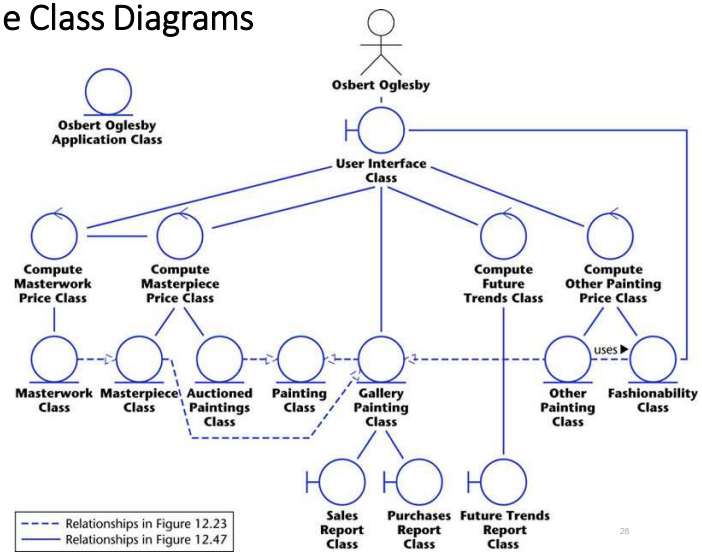
Incrementing the Class Diagram: The Osbert Oglesby Case Study

- In the course of realizing the various use cases
  - Interrelationships between classes become apparent
- Accordingly, we now combine the realization class diagrams



Sixth Iteration of the Class Diagrams

- Fifth iteration + realization class diagram



The Specification Document in the Unified Process

- The Unified Process is use-case driven
  - The use cases and the artifacts derived from them replace the traditional textual specification document
- The client must be shown each use case and associated artifacts, both diagrammatic and textual
  - These UML diagrams convey to the client more information more accurately than the traditional specification document
  - The set of UML diagrams can also play the same contractual role as the traditional specification document

The Specification Document (contd)

- A scenario is a specific execution sequence
- The client can therefore appreciate how the product works equally well from
  - A use case together with its scenarios, or
  - A rapid prototype
- The difference is
  - The use cases are successively refined, with more information added each time, whereas
  - The rapid prototype is discarded
- However, a rapid prototype of the user interface is required
  - Specimen screens and reports are needed (not a complete rapid prototype)

More on Actors and Use Cases

- To find the actors, consider every role in which an individual can interact with the software product
  - Example: **Applicants**, **Borrowers**
- Actors are not individuals
  - They are roles played by those individuals
- Find all the different roles played by each user
  - From the list of roles, extract the actors
- In the Unified Process
  - The term *worker* is used to denote a role played by an individual
  - In the Unified Process, **Applicants** and **Borrowers** are two different workers
- In common parlance
  - The word "worker" usually refers to an employee
- In this book, the word "role" is used in place of "worker"

More on Actors and Use Cases (contd)

- Within a business context, finding the roles is easy
  - They are displayed within the use-case business model
- To find the actors
  - Find the subset of the use-case business model that corresponds to the use-case model of the requirements
- To find the actors (in more detail):
  - Construct the use-case business model
  - Consider only those parts of the business model that correspond to the proposed software product
  - The actors in this subset are the actors we seek
- Within a business context, finding use cases is easy
- For each role, there will be one or more use cases
  - Find the actors (see previous slide)
  - The use cases then follow
- Challenges of the Object-Oriented Analysis Workflow
  - Do not cross the boundary into object-oriented design
  - Do not allocate methods to classes *yet*
    - Reallocating methods to classes during stepwise refinement is wasted effort