esiea
ECOLE D'INGENIEURS
DU MONDE NUMERIQUE

OBJECT-ORIENTED PROGRAMMING

INF3034

Pedro Braconnot Velloso

Class 2 — Project

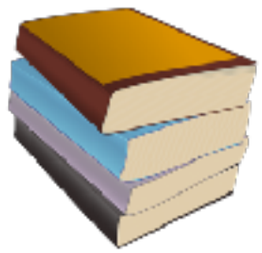
Agenda

- Project
  - Overview
  - Requirements
  - Team
  - Requirements specification
  - Organization
  - Evaluation

Project — Virtual store

Project overview

Virtual Store



Data Base

GUI

transaction

Buy

search

Client

Requirements

- Design and implementation of a desktop application
  - Graphical User Interfaces
  - Stock management
    - Show client list
    - Keep record of sales transaction
- The store sells books, DVDs, and games

Project Team

- Professor
  - Project owner — System Analyst
  - Software requirements specification
  - Specify the deliverables
  - Follow the project
- Students
  - Program development team
    - Implement the project considering the specifications
    - Deliver the deliverables in time

Software requirements specification

- Establishes the basis for an agreement
  - Between customers and contractors or suppliers
    - On how the software product should function

Deliverables

- A deliverable can be
  - A report
  - A document
  - A software product
  - A server upgrade
  - Any other building block of an overall project

Software requirements specification

GUI Specification

1. The user can view a product category list (MUST)
2. The category list presents an item << All >> (MUST)
3. The category list presents the following items: (MUST)
    1. DVDs
    2. Books
    3. Games
4. The category list is dynamically created from a list of product categories in a configuration file (SHOULD)
5. Product list (MUST)

GUI Specification

6. The product list is limited by the category the selected category (MUST)
7. The product list is sorted by price (SHULD)
8. The user can browse the list of products (MUST)

GUI Specification

9. The user can select a product:
    1. The description of the selected product will be displayed in a dedicated area (MUST)
    2. The name of the selected product will be displayed in a << Transaction >> area (MUST)
    3. The description of the selected product contains the info on stock depletion (SHOULD)
    4. The image of the selected product will be displayed in a dedicated area (SHOULD)
    5. Product image can be displayed as raster images (PNG, TIFF, JPEG) (SHOULD)

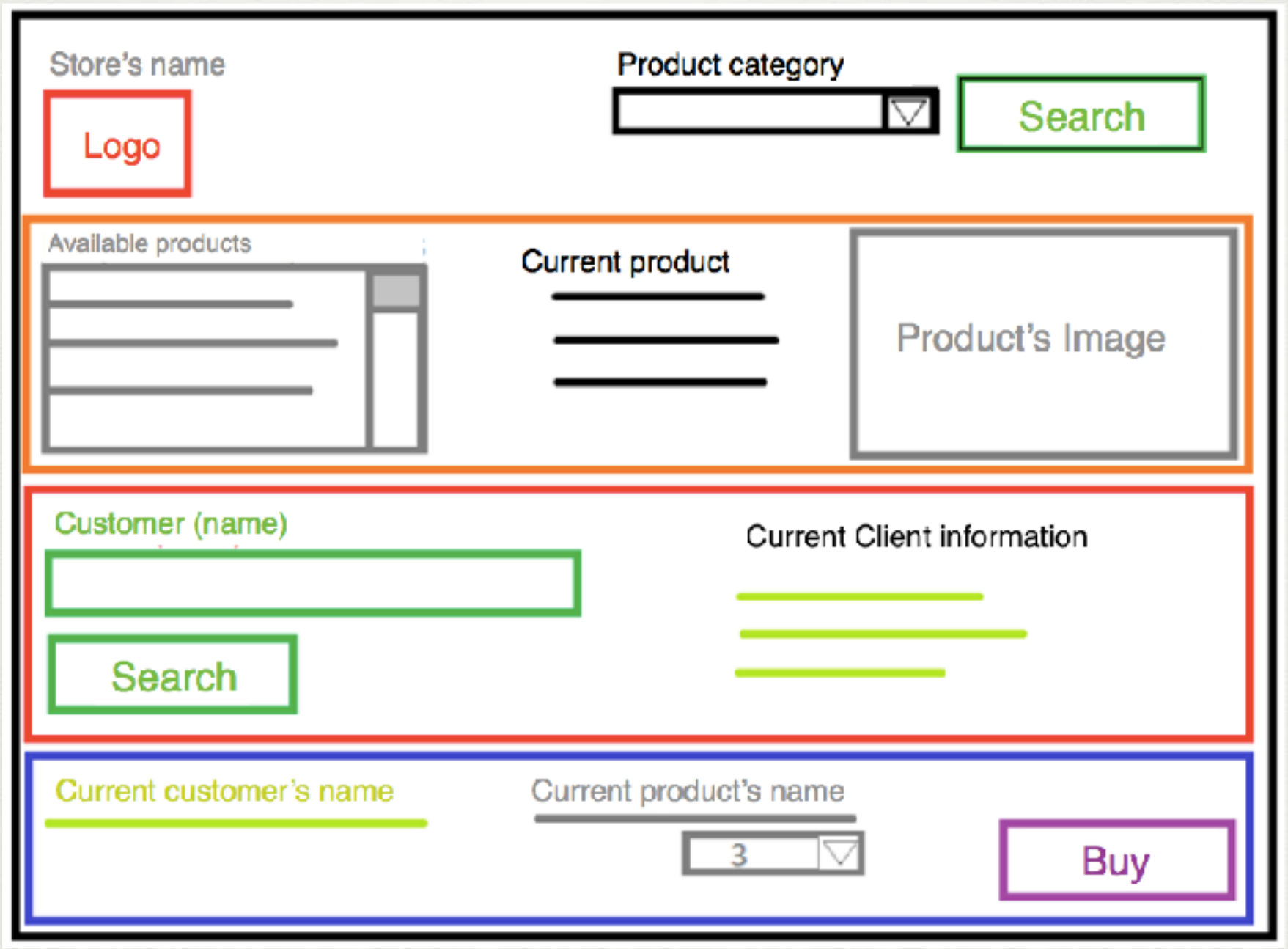
GUI Specification

10. The user can enter a customer name (MUST)
11. The user can search a customer in the list of customers (MUST)
12. If the customer already exists in the customer file, his information is displayed (MUST)
13. If the customer does not exist in the customer file, the user can create a new customer (last name, first name, address) (SHOULD)
14. The selected customer's name will be displayed in a <<Transaction>> area (MUST)

GUI Specification

15. The <<Transaction>> zone presents: (MUST)
    1. The name of the current product
    2. The name of the current customer
    3. The amount of products to buy
    4. A button to initiate the purchase process

GUI Model


Store's name
Logo
Product category
Search
Available products
Current product
Product's Image
Customer (name)
Search
Current Client information
Current customer's name
Current product's name
3
Buy

Product Specification

16. The product list is stored on disk as a .XML file (MUST)
17. The name of the file that contains the product is <<product.xm>> (MUST)
18. The file is in the <<files>> directory (MUST)
19. The file containing the image is in the "files" directory (SHOULD)
20. The categories are: (MUST)
    1. DVDs
    2. Books
    3. Games

Product Specification

21. All products have the following characteristics: (MUST)
    1. Category
    2. Name
    3. Price
    4. Unique Identification (ID)
    5. Number of products available in stock
    6. Descriptive image

Product Specification

22. A DVD presents the following characteristics: (MUST)
    1. Casting list
    2. Genre
        - Comedy, action, drama, horror, sci-fi, family, animation
    3. Duration
23. A book presents the following characteristics: (MUST)
    1. Author
    2. Language
        1. English, French, German, Spanish, Chinese
    3. Number of pages

Product Specification

24. A game presents the following characteristics: (MUST)
    1. Genre
        - arcade, adventure, role, multi-player, educational, strategy
    2. Platform
        1. Nintendo, PC, Playstation, Xbox

Customer Specification

25. The product list is stored on disk as a .XML file (MUST)
26. The name of the file that keeps information about a costumer is <<customer.xml>> (MUST)
27. The file is in the <<files>> directory (MUST)
28. Customer informations are: (MUST)
    1. Unique identifier
    2. Last Name
    3. First name
    4. Address

Transactions Specification

29. Each transaction will be saved on the disk in an .xml file which will contain all the transactions since the installation of the application (MUST)
30. The name of the file that contains the transactions is <<transactions.xml>> (MUST)
31. The file is in the <<files>> directory (MUST)
32. A transaction contains: (MUST)
    1. The unique identifier of the customer
    2. The unique product identifier
    3. The number of products
    4. Date / time
33. When a transaction is performed, the stock of the current product is updated, in the HMI and in the product file (MUST)

Store Specification

34. The store contains:
    1. Name (MUST)
    2. Logo (SHOULD)
35. The name of the file that contains the logo is <<logo.extension>> (SHOULD)
36. The file is in the <<files>> directory (SHOULD)
37. Logo image can be displayed as raster images (PNG, TIFF, JPEG) (SHOULD)

Startup and shutdown Specification

38. At startup, the application: (MUST)
    1. Read the list of products, which it must contain at least one product
    2. Read the customer list, which might be empty
    3. Launch and initialize the GUI
        1. Name and logo of the site, etc.
    4. Display the list of categories in the GUI
    5. Displays the product list in the GUI

Startup and shutdown Specification

39. When stopped, the application: (MUST)
    1. Closes the GUI
    2. Free all used resources (memory, disk)
40. After a non-predicted stop, no transaction is lost (MUST)

General Specification

41. The application must support other types of products and categories with a minimum of development effort (MUST)
42. The language supported by the application is the "US English" (MUST)
43. The javaDoc documentation is in English (MUST)
44. The documentation will contain the class diagrams in UML (SHOULD)
45. All classes will be tested with JUnit with no error detected (MUST)
46. The log systems: (MUST)
    1. The name of the log file is <<log.txt>>
    2. The log file will be created in the <<files>> directory
    3. The Log class supports three types of messages: Error, Warning, and Info

Deliverables

- Directory structure

```
project
   |bin | store | business (.class)
                  | gui     (.class)
                  | test    (.class)
                  | util    (.class)
|src | store | business (.java)
                  | gui     (.java)
                  | test    (.java)
                  | util    (.java)
|doc
|files
compile.bat
run.bat
test.bat
```

Organization

- 3 follow-up sessions of 1h30
- By group of 2 or 3 students
- Projects deadline
  - 10 December 2019
- On the course Moodle:
  - https://learning.esiea.fr/course/view.php?id=26

Must avoid!

- Plagiarism
- Procrastination
- Being absent at follow-up sessions
- Do not take advantage of follow-up sessions

Project evaluation

	Points
Implementation — MUST specification	7
Implementation — SHOULD specification	2
Object-Oriented aspects (abstraction, exceptions, errors management, collections	3
Individual tests	3
Code quality — respect programming basic rules	2
Deployment and package	2
JavaDoc documentation for all classes	1

Object design

- The design life of an object is not limited to the time when you're writing the program.
  - Five stages
- Object discovery
  - Initial analysis of a program.
  - Discovered by looking for
    - External factors and boundaries
    - Duplication of elements in the system
    - Commonality between classes suggesting base classes and inheritance may appear right away, or later in the design process

Object design

- Object assembly
  - As we are building an object
    - Discover the need for new members that didn't appear during discovery.
  - The internal needs of the object may require other classes to support it.

Object design

- System construction
  - More requirements for an object may appear at this later stage.
    - As we learn, we evolve our objects.
- The need for communication and interconnection with other objects in the system
  - May change the needs of your classes or require new classes

Object design

- System extension
  - Adding new features to a system
    - One may discover that your previous design does not support easy system extension
  - One can restructure parts of the system
    - possibly adding new classes or class hierarchies.

Object design

- Object reuse
  - This is the real test for a class
    - When someone tries to reuse it in an entirely new situation
      - Probably discover some shortcomings.
  - Changing a class to adapt to more new programs
    - The general principles of the class will become clearer, until you have a truly reusable type
  - Do not expect most objects from a system design to be reusable
    - It is perfectly acceptable for the bulk of your objects to be system-specific
    - Reusable types tend to be less common, and they must solve more general problems in order to be reusable

Guidelines for object design

- Let a specific problem generate a class, then let the class grow and mature during the solution of other problems
- Discovering the classes you need (and their interfaces) is the majority of the system design.
- Do not force yourself to know everything at the beginning
- Start programming; get something working so you can prove or disprove your design
- Always keep it simple
  - Little clean objects with obvious utility are better than big complicated interfaces

FYI

- Nasa Space Apps Challenge 2019
  - https://www.spaceappschallenge.org/
- International hackathon for coders, scientists, designers, storytellers, makers, builders, technologists, and others in cities around the world, where teams engage with NASA's free and open data to address real-world problems on Earth and in space

esiea
ECOLE D'INGENIEURS
DU MONDE NUMERIQUE

OBJECT-ORIENTED PROGRAMMING

INF3034

Pedro Braconnot Velloso

Class 2 — Project