Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative

SUPSI

Editor Image 2D

Studente/i
Riccardo Cristallo
Adel El Taher

Corso di laurea	Modulo / Codice Progetto	Anno
Bachelor Ingegneria Informatica	Lab di ingegneria del software II	2024/2025

Committente	Data
Giancarlo Corti	16/12/2024

Table of contents

Context and motivations

Topics Covered

- TeamWork
- Project Management Tools
- Project Structures

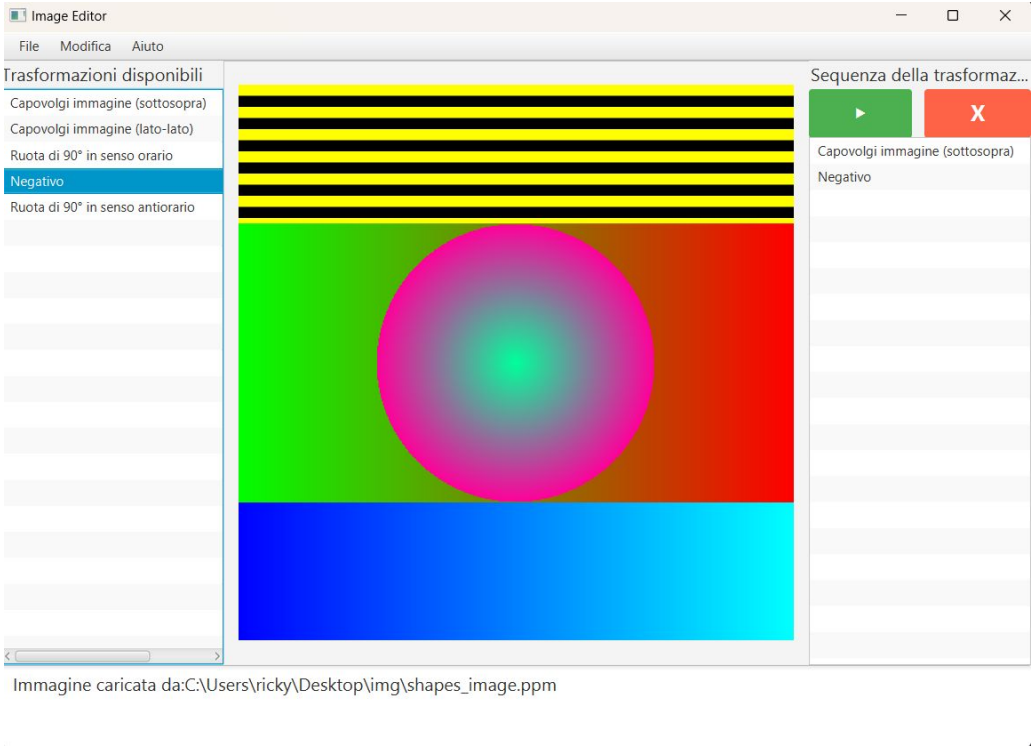
Context and Purpose of the Work

- Didactic: The didactic component will now contribute to the final grade with an adjustment of plus or minus 0.5 points

- Teamwork: The work must be carried out collaboratively as a team in order to help develop skills for effective teamwork following the Agile methodology of Software Engineering (SE)

Problem

- **Manual Implementation:** No external libraries for PNM handling, requiring custom readers, writers, and viewers.
- **Complex GUI Design:** Entire interface built from scratch, including menus, lists, and feedback log.
- **Testing Efforts:** Extensive unit tests and GUI tests required for all functionalities.



Use Case Diagram

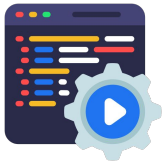
Team organization

Version Control: Used Git for collaboration and code versioning.

Agile Methodology: Followed iterative development with regular scrum meetings.

Requirement Elicitation: Gathered and refined information about app functionalities.

Project structure



MVC Architecture: Organized code into Model, View, and Controller for clarity and maintainability.

Layered Design: Separated backend logic and frontend UI into distinct modules.

Unit Testing: Ensured robustness of individual components.

End-to-End Testing: Validated full application workflows and user interactions.

Patterns used

Creational Patterns

- **Singleton**

Behavioural Pattern

- **Observer**
- **Strategy**
- **Template Method**

Observer Pattern

```java
public interface ObserverImage {
    void update(PixelView[][] pixels);
}
```

```java
public class ImageSubject {
    protected final List<ObserverImage> observerImages = new ArrayList<>();

    public void registerObserver(ObserverImage observerImage) {
        observerImages.add(observerImage);
    }

    public void notifyObservers(PixelView[][] pixels) {
        for (ObserverImage observerImage : observerImages) {
            observerImage.update(pixels);
        }
    }
}
```

```java
public class DisplayPNMImageView implements ObserverImage, DisplayPNMImageViewInterface {
    private static DisplayPNMImageView mySelf;

    @FXML
    private StackPane stackPane;

    // Other methods

    @Override
    public void update(PixelView[][] pixels) {
        int height = pixels.length;
        int width = pixels[0].length;
        stackPane.getChildren().clear();

        Canvas canvas = new Canvas(width, height);
        GraphicsContext gc = canvas.getGraphicsContext2D();


        for (int y = 0; y < height; y++) {
            for (int x = 0; x < width; x++) {
                gc.setFill(javafx.scene.paint.Color.rgb(
                        Math.min(255, Math.max(0, pixels[y][x].red())),
                        Math.min(255, Math.max(0, pixels[y][x].green())),
                        Math.min(255, Math.max(0, pixels[y][x].blue()))
                ));
                gc.fillRect(x, y, 1, 1);
            }
        }
        stackPane.getChildren().add(canvas);
    }
}
```

```java
public class LoadImageModel extends ImageSubject implements LoadImageModelInterface {
    private static LoadImageModel instance;

    // Other fields and methods ...

    @Override
    public boolean readImage(String filePath) {
        try {
            Pixel[][] pixels = loadImagePNMController.readImage(filePath);
            PixelView[][] pixelViews = convertToPixelView(pixels);
            notifyObservers(pixelViews);
            infoBarSubject.notifyObservers(resourceBundle.getString("label.imageLoaded") + filePath);
            return false;
        } catch (IllegalArgumentException e) {
            infoBarSubject.notifyObservers(e.getMessage() + ": " + filePath);
            return true;
        }
    }

    // ConvertToPixelView method ...

}
```

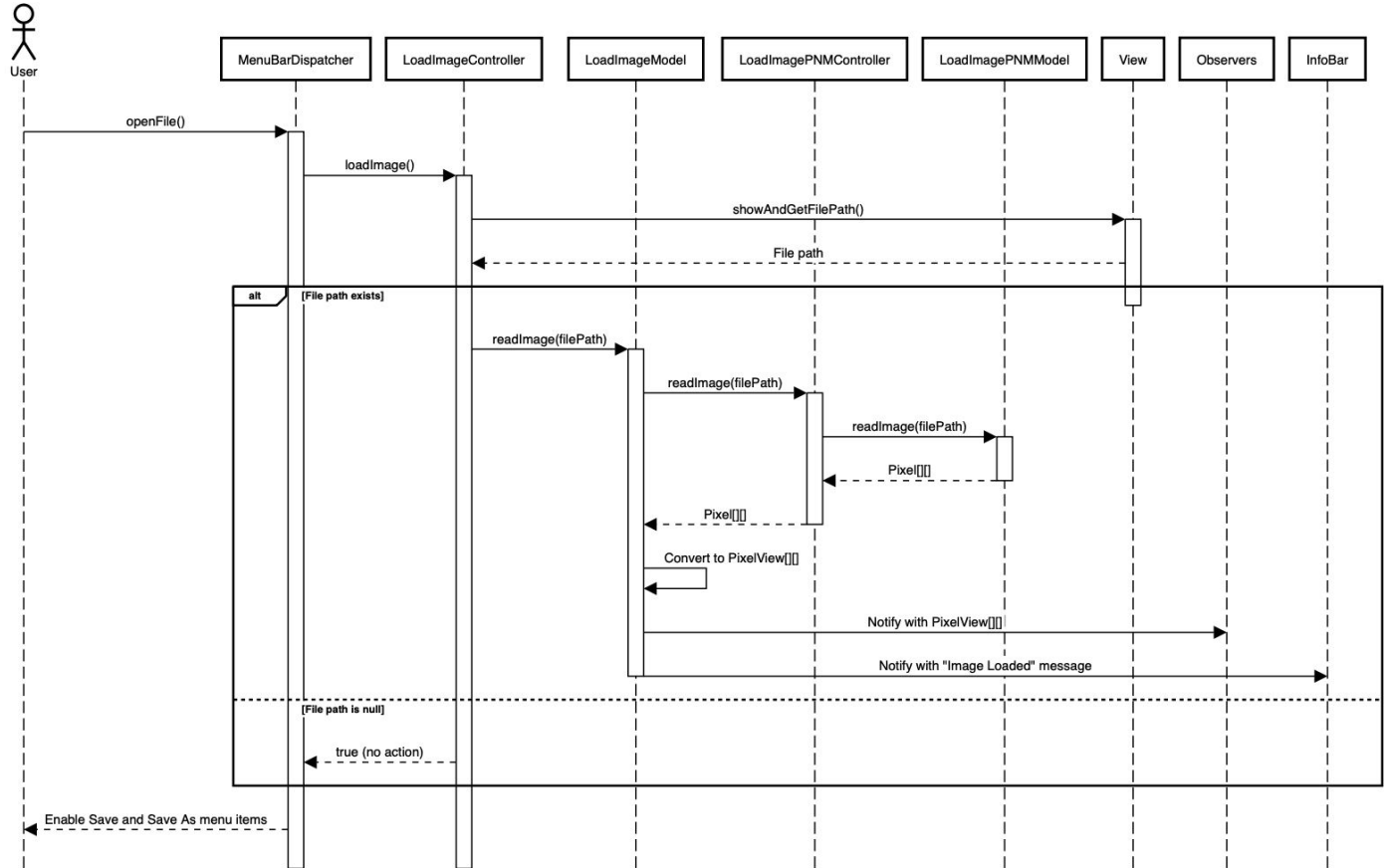
Strategy Pattern

```java
public class PNMObject {
    private String header;
    private int width;
    private int height;
    private int maxVal;
    private Pixel[][] pixels;
    private String filePath;
    private ExportInterface exportStrategy;

    /*
        Getter and Setter
        Methods
    */

    public void setExportStrategy(ExportInterface exportStrategy) {
        this.exportStrategy = exportStrategy;
    }

    public void export() {
        exportStrategy.export(this);
    }

}
```

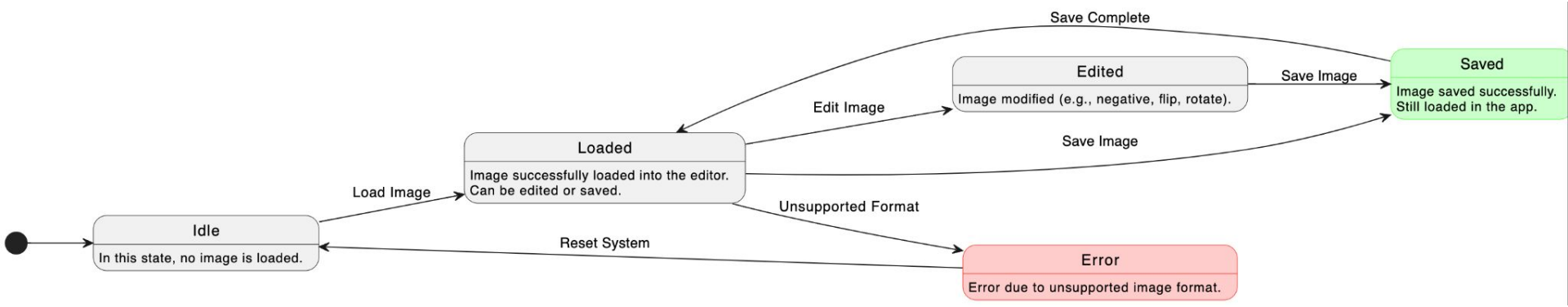
```java
public interface ExportInterface {
    void export(PNMObject obj);
}
```

```java
public class ExportFromPPMToPGM implements ExportInterface{
    @Override
    public void export(PNMObject obj) {
        if ("P3".equals(obj.getHeader())) {
            convertToGrayscale(obj);
        }
    }

    private void convertToGrayscale(PNMObject obj) {
        Pixel[][] pixels = obj.getPixels();
        for (int i = 0; i < obj.getHeight(); i++) {
            for (int j = 0; j < obj.getWidth(); j++) {
                Pixel px = pixels[i][j];
                int grayVal = (int) (0.299 * px.getRed() + 0.587 * px.getGreen() + 0.114 * px.getBlue());
                grayVal = Math.max(0, Math.min(255, grayVal));
                pixels[i][j] = new Pixel(grayVal);
            }
        }
        obj.setHeader("P2");
    }
}
```

Sequence diagram - Load image



State Machine Diagram



Approach

Methodology: Agile

- **Scrum Framework:**
  - Work organized into Iterations.
  - Weekly definition and assignment of Issues.
  - Regular Daily Stand-Ups and Sprint Reviews to track progress and adjust priorities.
- **Team Meetings**
  - Weekly group sessions every Saturday afternoon
  - Review of completed tasks.
  - Discussion of challenges and necessary adjustments.
  - Planning tasks for the upcoming week.

Approach

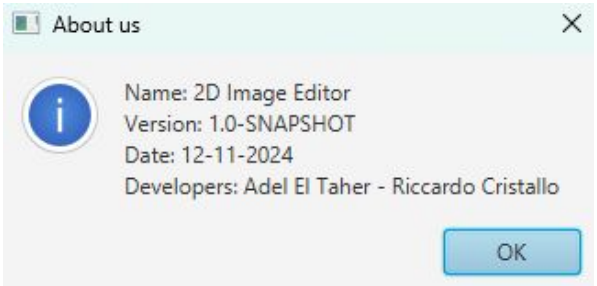
Language management and **continuous feedback** to the user to ensure clarity and understanding throughout the editing process, helping users make informed decisions and improving their experience.

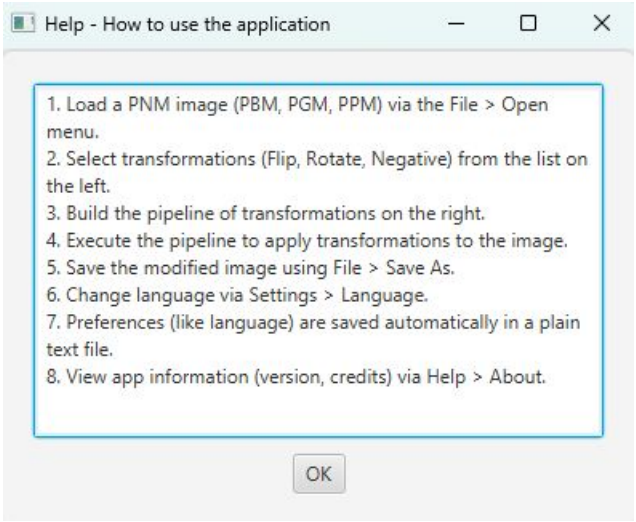
Image loaded from: /path/of/image
Transformation completed!
Image saved! *or* Image saved: /new/path/of/image
Select at least one transformation
Nothing to delete
Language saved!
Image not initialized
Loading errors: /path/of/image

Immagine caricata da: /percorso/immagine
Trasformazione completata!
Immagine salvata! *or* Immagine salvata: /nuovo/percorso/
Seleziona almeno una trasformazione
Niente da eliminare
Lingua salvata!
Immagine non inizializzata
Errori di caricamente: /percorso/immagine

Approach

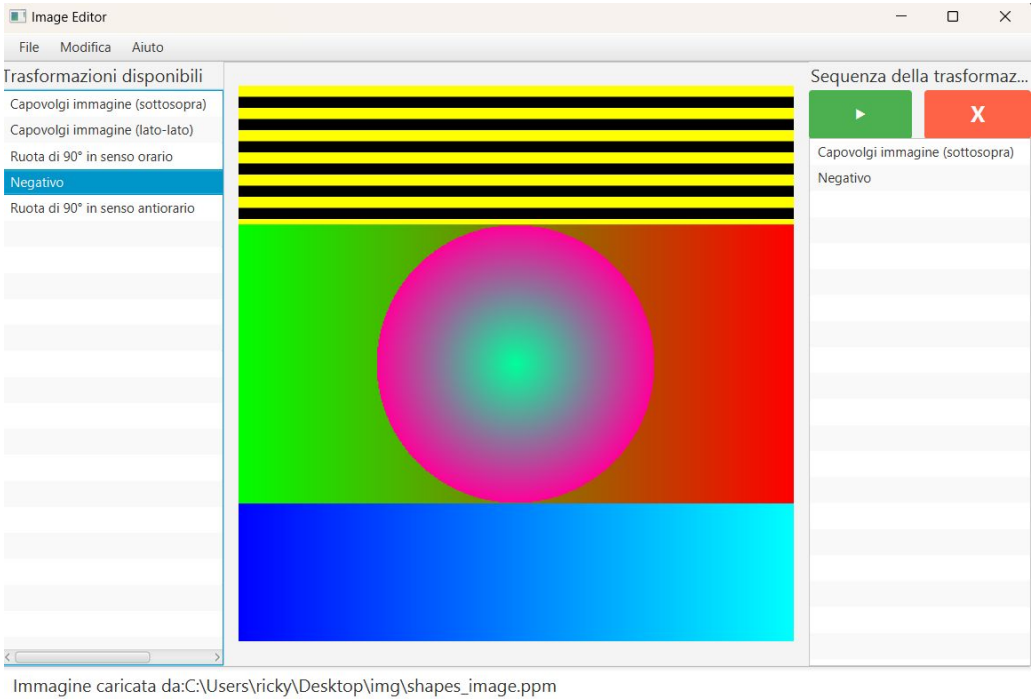
We have also implemented sections for **'About Us'** and **'Help'** to provide users with essential information about the app's functionality. These sections aim to guide users through the app and offer support for a better overall experience





Approach

Two pipelines, one showing the available transformations and the other showing the selected transformations. The last pipeline can be reset.

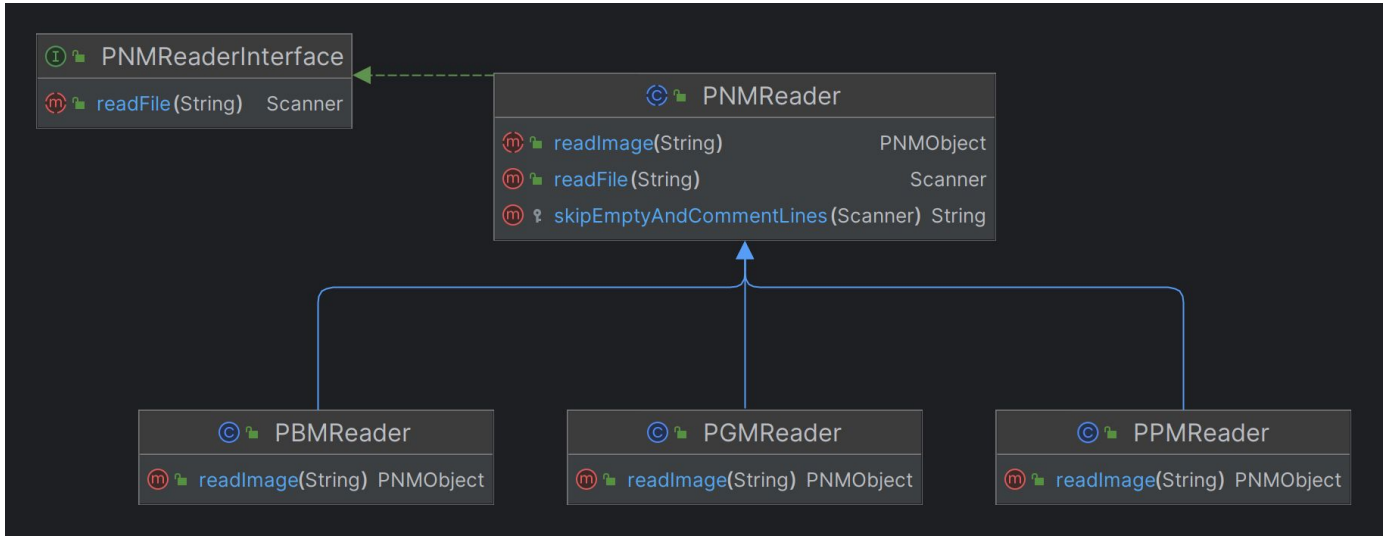


Approach

Reading and **writing** of **PNM** (Portable Any Map) formats.

We supported three formats: PBM, PGM, and PPM.

The design that we used:



Dynamic demo

Conclusions

The Agile methodology proved invaluable for organizing tasks and dividing work efficiently among the team. However, for shorter and simpler tasks, it sometimes became time-consuming.

One of our main challenges was when our teammate left the project at a critical moment, requiring us to quickly adapt and redistribute responsibilities.

Despite these difficulties, we successfully completed the project and are proud of the results we achieved!