UML: Activity & Sequence Diagrams

TP – Sequence & Activity Diagrams

Imane Fouad
Université Mohammed VI Polytechnique (UM6P)

By Imad Boutbaoucht

Contents

1. Exercise 1 – Repair Record Creation

Description

To create a repair record, the workshop manager enters the car search criteria into the system. The repair management software lists cars matching the criteria. If the car exists, the workshop manager selects it; otherwise, they enter the new vehicle's information. If the car is under warranty, the repair request date must be entered. Reception and return dates must always be entered. If the damage is covered by insurance, the manager selects the appropriate insurance. Finally, the software records the repair form.

Implementation Objective

You need to model the above process using an **Activity Diagram with partitions**, representing:

- The workshop manager's actions.
- The system's responses.
- Decision points (existence of car, warranty, insurance).
- Synchronization of actions before the final recording step.

Diagram

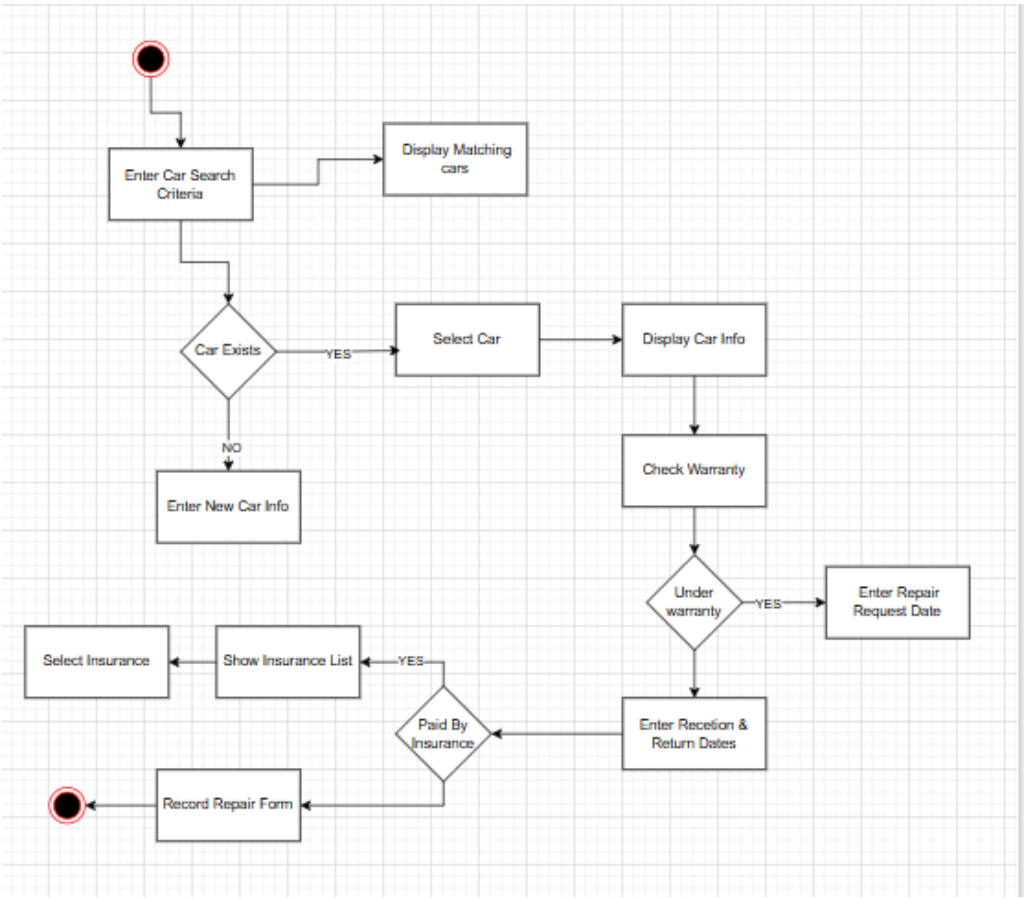

Figure 1.1: Activity Diagram – Repair Record Creation

2. Exercise 2 – Minesweeper Reveal Behavior

Description

When a player reveals a cell:

- If the cell contains a mine, the game ends immediately.
- If the cell contains a number, it is revealed and the game checks if the player has won.
- If the cell is empty, all neighboring cells are revealed recursively.

Implementation Objective

You must model this interaction using a **Sequence Diagram** that includes:

- The `Player`, `Game`, and `Cell` objects.
- Messages corresponding to revealing a cell, checking conditions, and recursive actions.

Diagram

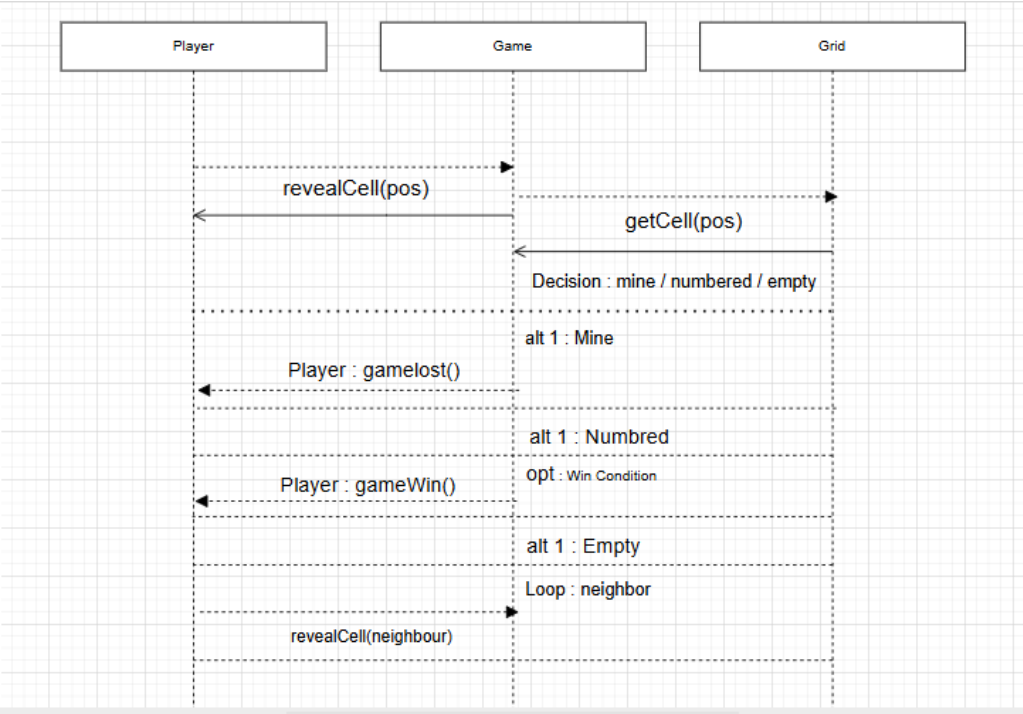

Figure 2.1: Sequence Diagram – Minesweeper Cell Reveal

3. Exercise 3 – Chocolate Mousse Recipe

Description

Recipe steps:

1. Break chocolate into pieces and melt it.
2. Concurrently, break the eggs and separate whites from yolks.
3. Once the chocolate is melted, add the yolks.
4. Whisk the egg whites until stiff.
5. Gently fold them into the chocolate mixture.
6. Pour into ramekins.
7. Chill for 3 hours before serving.

Implementation Objective

Represent this process using an **Activity Diagram** with concurrency. It should show:

- Parallel paths for melting chocolate and preparing eggs.
- Synchronization before mixing.
- Sequential steps leading to the final serving stage.

Diagram

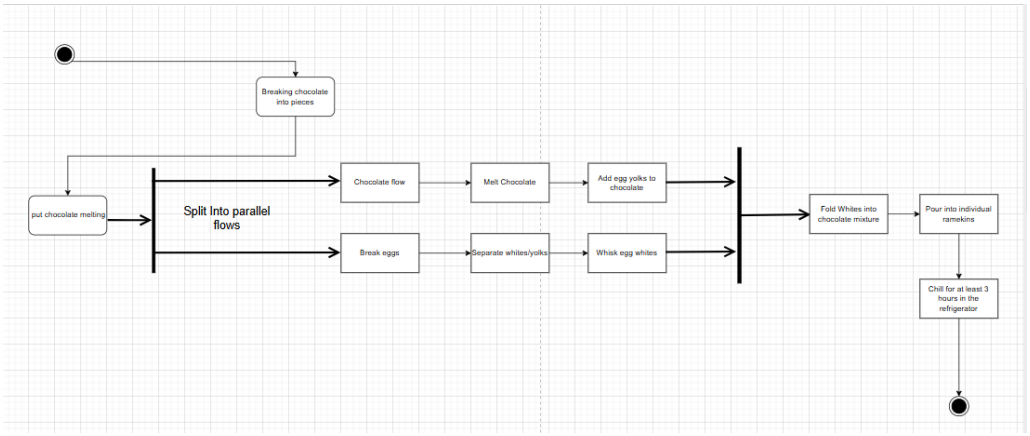

Figure 3.1: Activity Diagram – Chocolate Mousse Recipe

4. Exercise 4 – Library Management Use Cases

Description

We analyze two scenarios: **Check Overdue** and **Borrow a Book**.

Scenario 1 – Check Overdue

1. The system retrieves all loans for the member.
2. For each loan, it checks if the return date has passed.
3. If overdue, the system sets the member's status to "Suspended".

Scenario 2 – Borrow a Book

1. The librarian selects the borrow function.
2. The system checks if the member is allowed to borrow.
3. It checks for overdue books and updates the status if necessary.
4. It verifies if the book is available.
5. Creates a new loan, marks the book as unavailable, and increments the member's loan count.
6. Updates the member's status.

Implementation Objective

You must model both use cases with **Sequence Diagrams**, showing:

- Interactions between the `Librarian`, `System`, `Member`, and `Book`.
- Control flows for overdue checking and book borrowing.

Diagrams

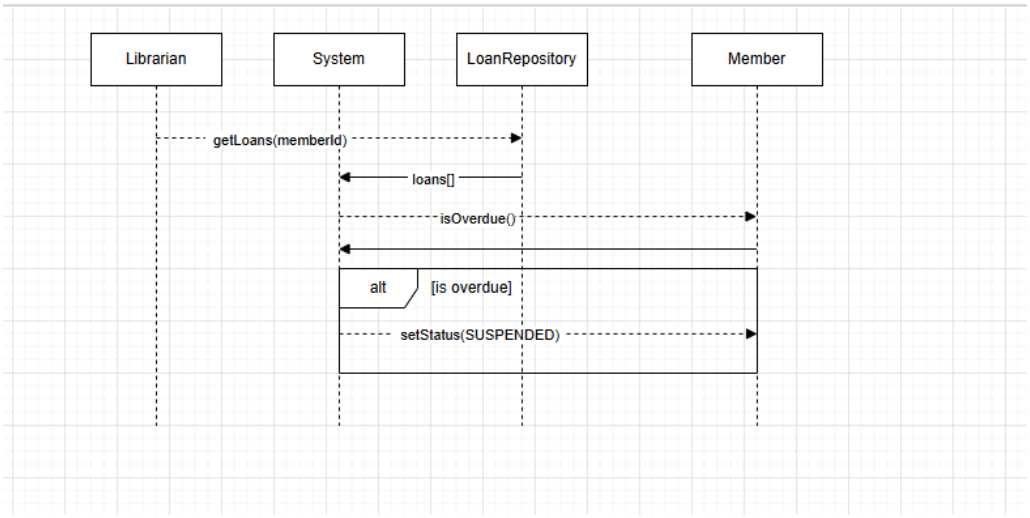

Figure 4.1: Sequence Diagram – Check Overdue

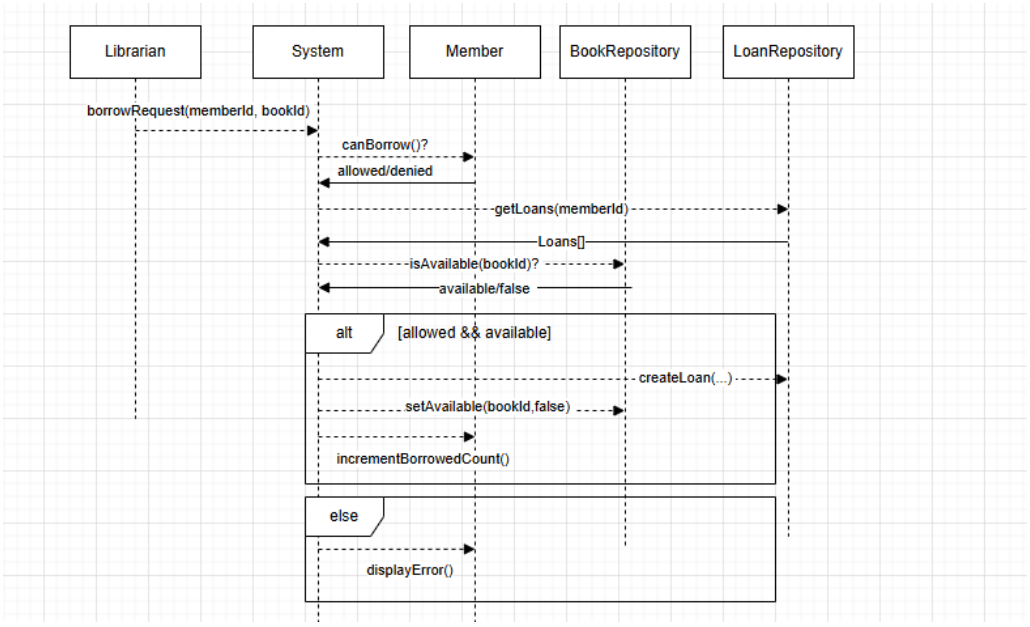

Figure 4.2: Sequence Diagram – Borrow a Book