Diagramme d’use-case

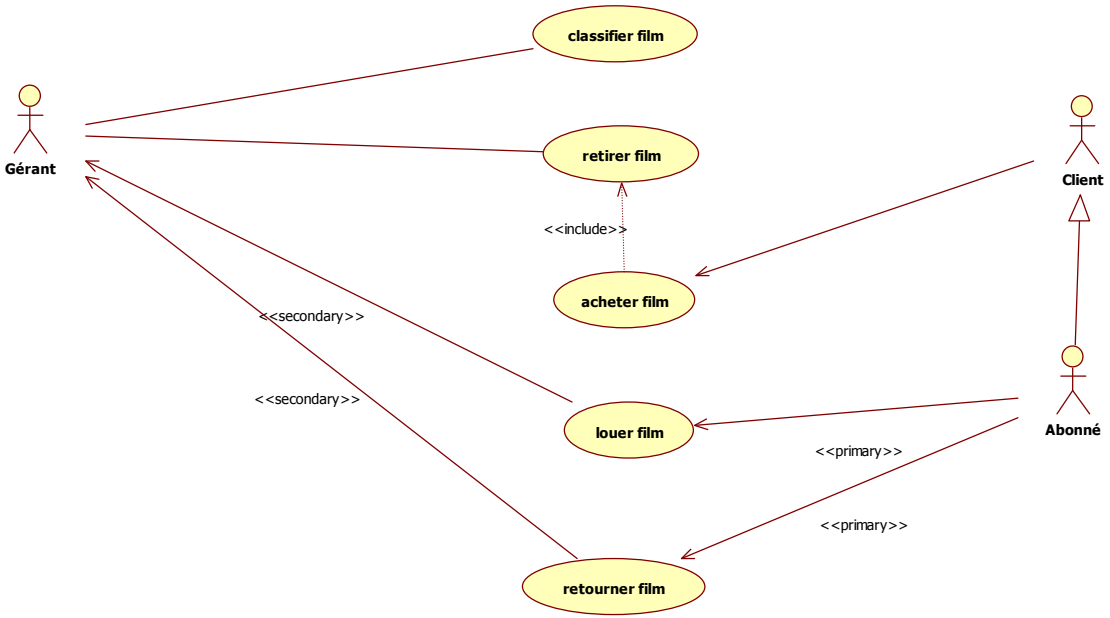


Diagramme de classes

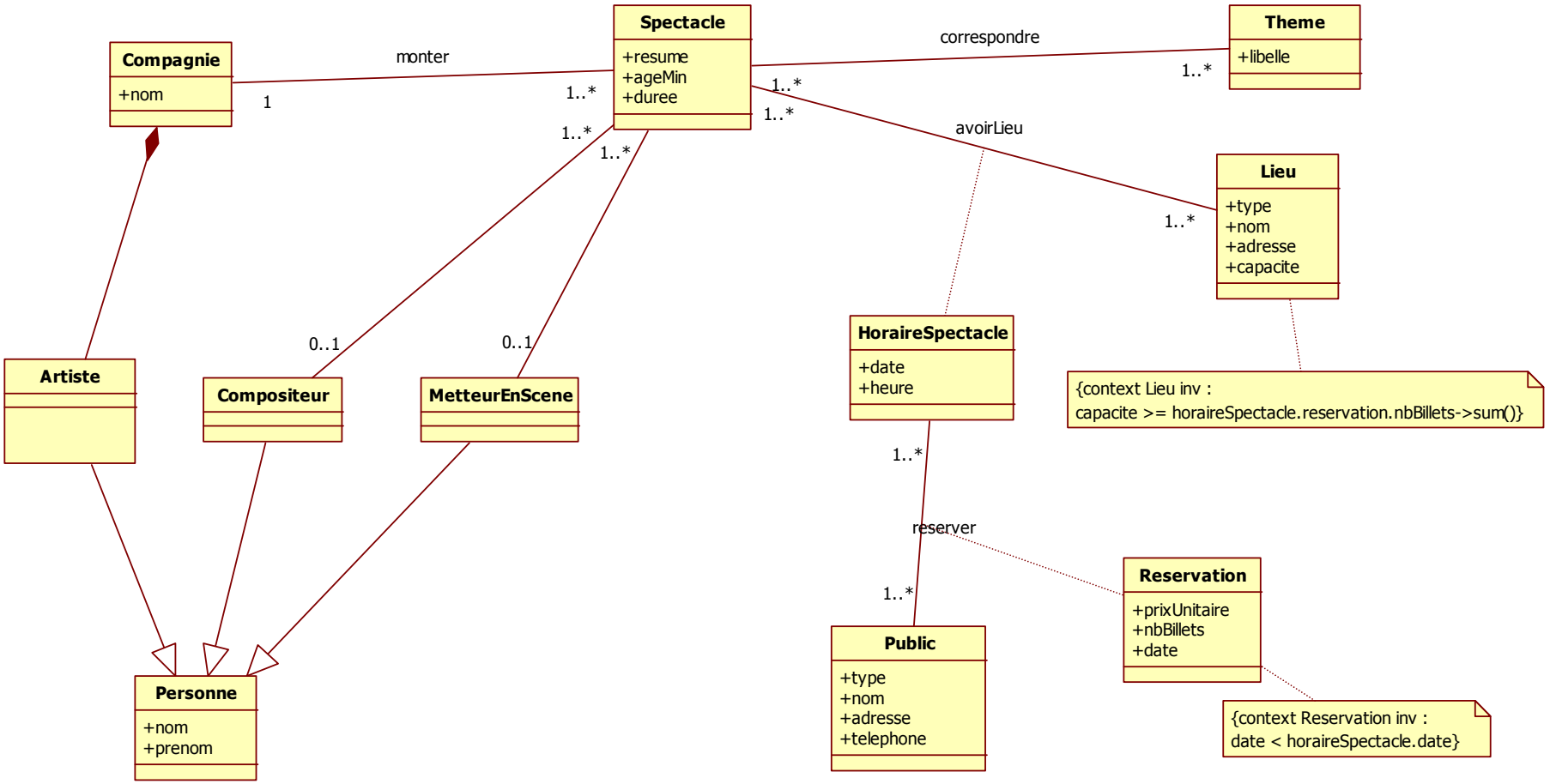
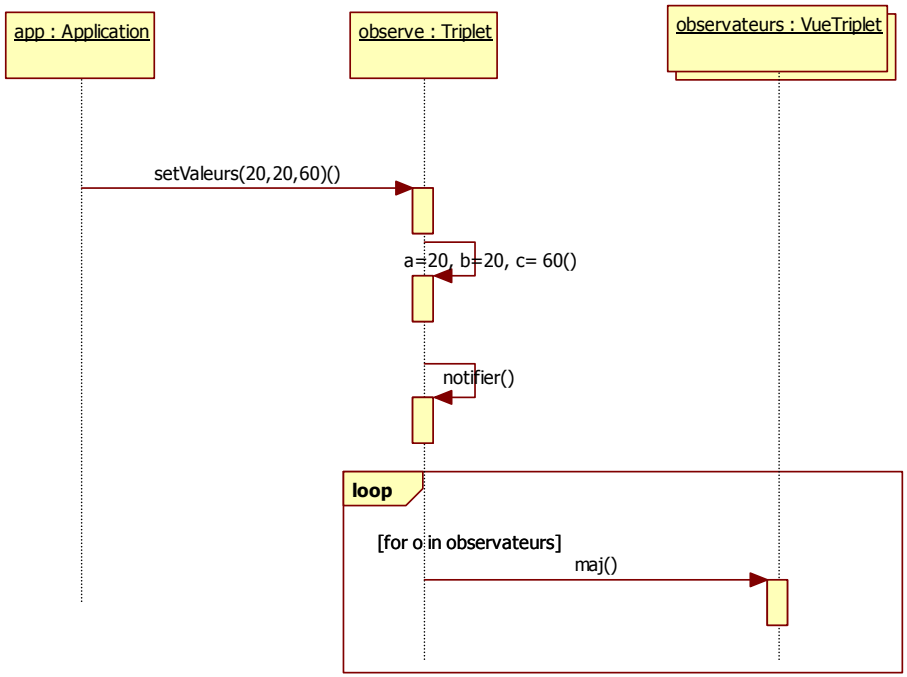
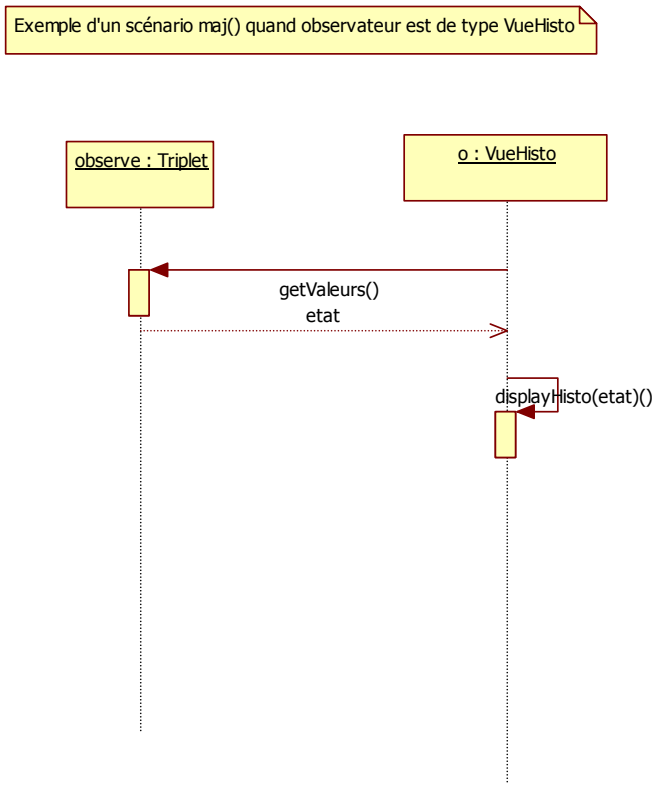


Diagramme de séquences :



Mapping UML Java :

```
/***************** Person.java*********************************/
import java.util.*;

public class Person {

      public enum Gender {male, female}

      /* attributs propres à la classe */

      private Gender gender;
      private boolean isMarried;
      private boolean isUnemployed ;
      private Date birthday;
      private int age;
      private String firstName;
      private String lastName;

      /* attributs d'association */

      private Vector<Company> managedCompanies;
      private Vector<Company> employers;
      private Person[] parent;                 // array's size = 2
      private Vector<Person> child;
      private Person spouse;                   // wife or husband

      /* pour pouvoir calculer la revenue, il faut associer une personne avec ses jobs */

      private Vector<Job> jobs ;

      // constructeur

      public Person(Gender g, Date bd, String fn, String ln, Person pere, Person mere) {
            gender = g;
            isMarried = false;
            isUnemployed = true;
            birthday = bd;
            //age = Date.now() - birthday;
```

```
            firstName = fn;
            lastName = ln;
            managedCompanies = new Vector<Company>();
            employers = new Vector<Company>();
            parent = new Person[2];
            parent[0] = pere;
            parent[1] = mere;
            child = new Vector<Person>();
            jobs = new Vector<Job>();
      }

      public void addSpouse(Person p) {
            spouse = p;
            isMarried = true;
      }

      public void addEmployer(Company c) {
            if (isUnemployed)
                  isUnemployed = false;
            employers.add(c);
      }

      public float income(){
            float res = 0 ;
            for (Job j : jobs)
                  res += j.getSalary();
            return res;
      }

      /* d'autres méthodes éventuelles : getteurs, setteurs, addChild, addManagedCcompanies, ... */
}

/***************** Company.java*********************************/

import java.util.*;

public class Company {
      /* attributs propres à la classe */

      private String name;
```

```
	private int numberOfEmployees;

	/* attributs d'association */

	private Person manager;
	private Vector<Person> employees;

	// constructeur
	public Company(String n, Person m) {
		name = n;
		manager = m;
		numberOfEmployees = 1;
		employees = new Vector<Person>();
	}

	public void addEmployee(Person p) {
		employees.add(p);
		numberOfEmployees ++;
	}
}

/***************** Job.java*********************************/

import java.util.*;

public class Job {
	/* attributs propres à la classe */

	private String title;
	private Date startDate;
	private float salary;

	/* attributs d'association */

	private Company employer;
	private Person employee;

	// constructeur

	public Job(String t, Date sd, float s, Company c, Person p) {
```

```
            title = t;
            startDate = sd;
            salary = s;
            employer = c;
            employee = p;
      }
      public float getSalary() {
            return salary;
      }

}

/***************** Marriage.java*********************************/

import java.util.*;

public class Marriage {

      /* attributs propres à la classe */

      private String place;
      private Date date;

      /* attributs d'association */

      private Person wife;
      private Person husband;

      // constructeur
      public Marriage(String p, Date d, Person w, Person h) {
            place = p;
            date = d;
            wife = w;
            husband = h;
      }
}
```