Université Bordeaux I
U.F.R. Mathématiques et Informatique

IUP MIAGe 3^{eme} année
Année 2003-2004

Lundi 12 janvier 2004, 10 heures - 11 heures 30
Tous documents autorisés.

Programmation par Objets
Alain Griffault

Numéro d'anonymat :

Introduction

Une personne naît, puis meurt. Durant sa vie, elle a pu se marier, divorcer et devenir veuf ou veuve. La figure 1 représente toutes les évolutions possibles au cours de l'existence d'une personne.

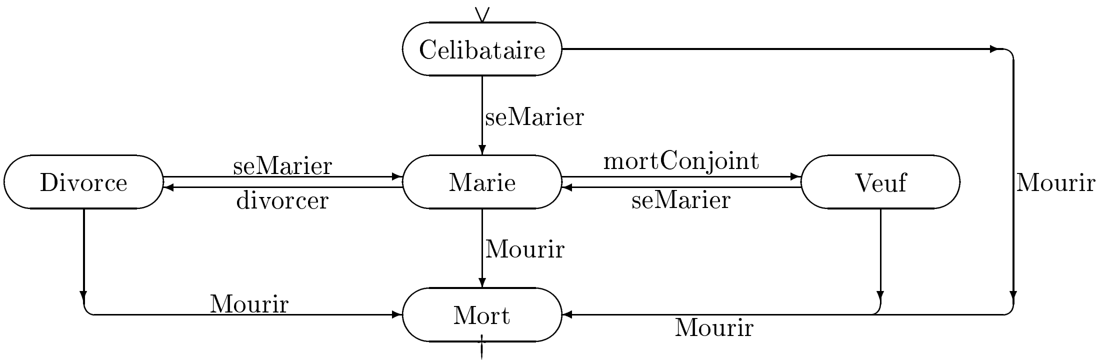

Fig. 1 – Le cycle de la vie

Le sujet porte sur la gestion (simplifiée) d'un groupe de personnes. L'étude fait évoluer une première version par l'utilisation de modèles de conception, puis de `Thread`.

1 Les étapes de la vie

Quatre personnes naissent, puis on affiche éventuellement trois étapes de leurs vies.
La classe `Personne` :

```java
import java.util.*;
import java.text.*;
public class Personne{
    private static int numero = 0;
    private static Random alea = new Random();
    private static SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
    private int ident;
    private Date naissance, deces;
    private boolean masculin;
    private boolean celibataire, marie, divorce, veuf, mort;

    public Personne(boolean garcon){
        ident = numero++;
        naissance = new Date();
        deces = null;
        masculin = garcon;
```

```java
        celibataire = true;
        marie = false;
        divorce = false;
        veuf = false;
        mort = false;
    }

    public String toString(){
        String res ="";
        if (masculin) {
            res +="Monsieur "+ident+", ne le ";
            res += sdf.format(naissance)+", est";
            if (celibataire) res +=" celibataire.\n";
            if (marie)       res +=" marie.\n";
            if (divorce)     res +=" divorce.\n";
            if (veuf)        res +=" veuf.\n";
            if (mort)        res +=" mort le "+sdf.format(deces)+".\n";
        }
        else {
            if (celibataire){
                res +="Mademoiselle "+ident+", est nee le ";
                res += sdf.format(naissance)+".\n";
            }
            else{
                res +="Madame "+ident+", nee le ";
                res += sdf.format(naissance)+", est";
            }
            if (marie)       res +=" mariee.\n";
            if (divorce)     res +=" divorcee.\n";
            if (veuf)        res +=" veuve.\n";
            if (mort)        res +=" morte le "+sdf.format(deces)+".\n";
        }
        return res;
    }

    public void mourir() throws Exception{
        if (mort)
            throw new Exception("Impossible");
        else {
            deces = new Date();
            celibataire = false;
            marie = false;
            divorce = false;
            veuf = false;
            mort = true;
        }
    }

    public void seMarier() throws Exception{
        if (mort || marie)
            throw new Exception("Impossible");
        else {
            celibataire = false;
            marie = true;
            divorce = false;
            veuf = false;
            mort = false;
```

```
        }
    }

    public void divorcer() throws Exception{
        if (mort || celibataire || veuf || divorce)
            throw new Exception("Impossible");
        else {
            celibataire = false;
            marie = false;
            divorce = true;
            veuf = false;
            mort = false;
        }
    }

    public void mortConjoint() throws Exception{
        if (mort || celibataire || veuf || divorce)
            throw new Exception("Impossible");
        else {
            celibataire = false;
            marie = false;
            divorce = false;
            veuf = true;
            mort = false;
        }
    }

    public void vieillir() {
        try {Thread.sleep(2000);}       // 2 secondes d'endormissement
        catch(InterruptedException ie){}
        finally {
            int a = alea.nextInt(4);
            try {
                switch (a) {
                case 0 : mourir(); break;
                case 1 : seMarier(); break;
                case 2 : divorcer(); break;
                case 3 : mortConjoint();
                }
                System.out.print(this);
            }
            catch (Exception e){} // Exceptions lors de chgt de situation
        }
    }

}
```

La classe Test :

```
import java.util.*;
public class Test{
    public static void main(String[] args){
        Personne[] personne = new Personne[4];
        Random alea = new Random();
        System.out.println("\tNaissances");
        for (int i=0; i<personne.length; i++){
            personne[i] = new Personne(alea.nextBoolean()); //Homme ou Femme
            System.out.print(personne[i]);
        }
```

```
        System.out.println("\tQuelques etapes de la vie");
        for (int time=0; time<3; time++)
            for (int i=0; i<personne.length; i++)
                personne[i].vieillir();
    }
}
```

et un exemple de sortie générée.

```
        Naissances
Mademoiselle 0, est nee le 11/01/2004 17:02:19.
Mademoiselle 1, est nee le 11/01/2004 17:02:19.
Monsieur 2, ne le 11/01/2004 17:02:19, est celibataire.
Monsieur 3, ne le 11/01/2004 17:02:19, est celibataire.
        Quelques etapes de la vie
Madame 0, nee le 11/01/2004 17:02:19, est mariee.
Madame 1, nee le 11/01/2004 17:02:19, est morte le 11/01/2004 17:02:31.
Monsieur 2, ne le 11/01/2004 17:02:19, est marie.
Monsieur 2, ne le 11/01/2004 17:02:19, est divorce.
Monsieur 3, ne le 11/01/2004 17:02:19, est mort le 11/01/2004 17:02:43.
```

2 Le modèle de conception *état*

Le modèle **Etat** permet à un objet de modifier son comportement, quand son état interne change. Tout se passera comme si l'objet changeait de classe.

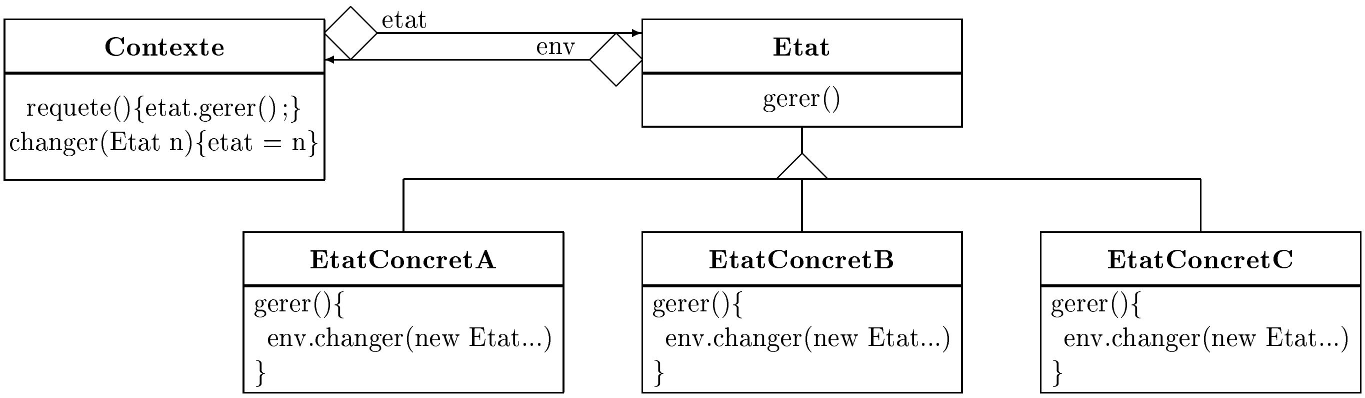

FIG. 2 – La structure du modèle État

L'objectif de l'exercice est que le même programme `test` génère une sortie équivalente.
Pour cela, le modèle **Etat** est implémenté non pas comme un interface, mais comme la classe `Situation` donnant les comportements par défauts.

```
import java.text.*;
public class Situation{
    public static SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
    Personne moi;

    public Situation(Personne me){moi = me;}

    public void mourir()       throws Exception{throw new Exception("Impossible");}
    public void seMarier()     throws Exception{throw new Exception("Impossible");}
    public void divorcer()     throws Exception{throw new Exception("Impossible");}
    public void mortConjoint() throws Exception{throw new Exception("Impossible");}
}
```

Question 2.1 (4 points) *Complétez (lignes 12-16, 33-36 et 45) le code de la classe* `Personne`.

```
1   import java.util.*;
2   public class Personne{
3       private static int numero = 0;
4       private static Random alea = new Random();
5
6       private int ident;
7       private Date naissance, deces;
8       private boolean masculin;
9       private Situation situation; // remplace les booleens.
10
11      public Personne(boolean garcon){
12
13
14
15
16
17      }
18
19      public int     getIdent()    {return ident;}
20      public boolean isMasculin()  {return masculin;}
21      public Date    getNaissance(){return naissance;}
22      public Date    getDeces()    {return deces;}
23      public String  toString()    {return situation.toString();}
24      public void    setDeces()    {deces = new Date();}
25
26      public void vieillir() {
27          try {Thread.sleep(2000);}
28          catch(InterruptedException ie){}
29          finally {
30              int a = alea.nextInt(4);
31              try {
32                  switch (a) {
33                  case 0 :                                  break;
34                  case 1 :                                  break;
35                  case 2 :                                  break;
36                  case 3 :
37                  }
38                  System.out.print(this);
39              }
40              catch (Exception e){} // Exceptions lors de chgt de situation
41          }
42      }
43
44      public void changerSituation(Situation sit){
45
46      }
47
48  }
```

Question 2.2 (3 points) *Complétez (lignes 5, 21 et 25) le code de la classe* `Divorce`.

```
1   import java.util.*;
2   import java.text.*;
3   public class Divorce extends Situation{
4       public Divorce(Personne me){
5
6       }
7       public String toString(){
8           String res ="";
9           if (moi.isMasculin()){
10              res +="Monsieur "+moi.getIdent()+", ne le ";
11              res += sdf.format(moi.getNaissance())+", est divorce.\n";
12          }
13          else{
14              res +="Madame "+moi.getIdent()+", nee le ";
15              res += sdf.format(moi.getNaissance())+", est divorcee.\n";
16          }
17          return res;
18      }
19
20      public void mourir() throws Exception{
21
22      }
23
24      public void marier() throws Exception{
25
26      }
27
28  }
```

3 Le modèle de conception *état* avec historique

On veut en plus pouvoir afficher les situations des différentes personnes à une date donnée. La classe `Test` est légèrement modifiée à cette fin (lignes 7 et 18-24).

```
1   import java.text.*;
2   import java.util.*;
3   public class Test{
4       public static void main(String[] args){
5           Personne[] personne = new Personne[4];
6           Random alea = new Random();
7           Date debut = new Date();
8           System.out.println("\tNaissances");
9           for (int i=0; i<personne.length; i++){
10              personne[i] = new Personne(alea.nextBoolean()); //Homme ou Femme
11              System.out.print(personne[i]);
12          }
13          System.out.println("\tQuelques etapes de la vie");
14          for (int time=0; time<3; time++){
15              for (int i=0; i<personne.length; i++)
16                  personne[i].vieillir();
17          }
18          Date fin = new Date();
19          Date milieu = new Date((debut.getTime()+fin.getTime())/2);
20          SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
21          System.out.println("\tSituation a la date "+sdf.format(milieu));
22          for (int i=0; i<personne.length; i++){
```

```
            System.out.print(personne[i].getSituation(milieu));
        }
    }
}
```

et un exemple de sortie générée.

```
        Naissances
Mademoiselle 0, nee le 11/01/2004 19:39:49.
Mademoiselle 1, nee le 11/01/2004 19:39:49.
Monsieur 2, ne le 11/01/2004 19:39:49, est celibataire.
Mademoiselle 3, nee le 11/01/2004 19:39:49.
        Quelques etapes de la vie
Madame 0, nee le 11/01/2004 19:39:49, est morte le 11/01/2004 19:39:51.
Madame 1, nee le 11/01/2004 19:39:49, est mariee depuis le 11/01/2004 19:39:53.
Monsieur 2, ne le 11/01/2004 19:39:49, est mort le 11/01/2004 19:39:55.
Madame 1, nee le 11/01/2004 19:39:49, est divorcee depuis le 11/01/2004 19:40:01.
        Situation a la date 11/01/2004 19:40:01
Madame 0, nee le 11/01/2004 19:39:49, est morte le 11/01/2004 19:39:51.
Madame 1, nee le 11/01/2004 19:39:49, est mariee depuis le 11/01/2004 19:39:53.
Monsieur 2, ne le 11/01/2004 19:39:49, est mort le 11/01/2004 19:39:55.
Mademoiselle 3, nee le 11/01/2004 19:39:49.
```

Il faut gérer pour chaque personne un historique des ses situations. Une solution consiste à associer à chaque nouvelle situation une date.

Question 3.1 (3 points) *Complétez (lignes 9-10, 14 et 18) le code de la classe* `Situation`.

```
1   import java.text.*;
2   import java.util.*;
3   public class Situation{
4       public static SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
5       Date debut;  // pour memoriser le debut de cette situation
6       Personne moi;
7
8       public Situation(Personne me){
9
10
11      }
12
13      public Date getDebut(){
14
15      }
16
17      public Date getDeces(){
18
19      }
20
21      public void mourir()       throws Exception{throw new Exception("Impossible");}
22      public void seMarier()     throws Exception{throw new Exception("Impossible");}
23      public void divorcer()     throws Exception{throw new Exception("Impossible");}
24      public void mortConjoint() throws Exception{throw new Exception("Impossible");}
25  }
```

Question 3.2 (4 points) *Complétez (lignes 12-15, 31-34, 43-44 et 48-54) le code de la classe* `Personne`.

```
1   import java.util.*;
2   public class Personne{
3       private static int numero = 0;
4       private static Random alea = new Random();
5
6       private int ident;
7       private boolean masculin;
8       private List vie = new ArrayList(); // pour gerer l'historique
9       private Situation situation;
10
11      public Personne(boolean garcon){
12
13
14
15
16      }
17
18      public int     getIdent()    {return ident;}
19      public boolean isMasculin()  {return masculin;}
20      public Date    getNaissance(){return ((Situation)vie.get(0)).getDebut();}
21      public Date    getDeces()    {return situation.getDeces();}
22      public String  toString()    {return situation.toString();}
23
24      public void vieillir() {
25          try {Thread.sleep(2000);}
26          catch(InterruptedException ie){}
27          finally {
28              int a = alea.nextInt(4);
29              try {
30                  switch (a) {
31                  case 0 :                                    break;
32                  case 1 :                                    break;
33                  case 2 :                                    break;
34                  case 3 :
35                  }
36                  System.out.print(this);
37              }
38              catch (Exception e){} // Exceptions lors de chgt de situation
39          }
40      }
41
42      public void changerSituation(Situation sit){
43
44
45      }
46
47      public Situation getSituation(Date evt){//retrouve et affiche une situation
48
49
50
51
52
53
54
55      }
56  }
```

Question 3.3 (1 point) *Complétez (ligne 10) le code de la classe* `Mort`*.*

```
1   import java.util.*;
2   import java.text.*;
3   public class Mort extends Situation{
4
5       public Mort(Personne me){
6           super(me);
7       }
8
9       public Date getDeces(){
10
11      }
12
13      public String toString(){
14          String res ="";
15          if (moi.isMasculin()){
16              res +="Monsieur "+moi.getIdent()+", ne le ";
17              res += sdf.format(moi.getNaissance())+", est mort le ";
18          }
19          else{
20              res +="Madame "+moi.getIdent()+", nee le ";
21              res += sdf.format(moi.getNaissance())+", est morte le ";
22          }
23          res += sdf.format(debut)+".\n";
24          return res;
25      }
26
27  }
```

Question 3.4 (1 point) *Expliquez comment vous pourriez réaliser une méthode* `undo` *à partir de la solution précédente.*

4 Le modèle de conception *état*, historique et Thread

Le défaut majeur de la solution précédente est la gestion du vieillissement par la classe `Test`. On souhaite maintenant que chaque personne soit assez indépendante dans son rythme de vie.
Une solution consiste à considérer les objets `Personne` comme des processus indépendants, c'est à dire des `Thread` java, toutes les autres classes étant inchangées. L'avantage immédiat est la possibilité pour la classe `Test` d'attendre la mort des personnes au lieu de limiter à priori le nombre d'étapes dans la vie de chacun.
La nouvelle classe `Test` (lignes 13-19 modifiées).

```
1   import java.text.*;
2   import java.util.*;
3   public class Test{
4       public static void main(String[] args){
5           Personne[] personne = new Personne[4];
6           Random alea = new Random();
7           Date debut = new Date();
8           System.out.println("\tNaissances");
9           for (int i=0; i<personne.length; i++){
10              personne[i] = new Personne(alea.nextBoolean()); //Homme ou Femme
11              System.out.print(personne[i]);
12          }
13          System.out.println("\tToutes les etapes de la vie");
14          for (int i=0; i<personne.length; i++)
15              personne[i].start();                //demarrer les Thread
16          for (int i=0; i<personne.length; i++){
17              try {personne[i].join();}           //attendre la fin des Thread
18              catch(InterruptedException ie){}
19          }
20          Date fin = new Date();
21          Date milieu = new Date((debut.getTime()+fin.getTime())/2);
22          SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
23          System.out.println("\tSituation a la date "+sdf.format(milieu));
24          for (int i=0; i<personne.length; i++){
25              System.out.print(personne[i].getSituation(milieu));
26          }
27      }
28  }
```

et un exemple de sortie générée.

```
        Naissances
Monsieur 0, ne le 11/01/2004 21:41:44, est celibataire.
Mademoiselle 1, nee le 11/01/2004 21:41:44.
Monsieur 2, ne le 11/01/2004 21:41:44, est celibataire.
Monsieur 3, ne le 11/01/2004 21:41:44, est celibataire.
        Toutes les etapes de la vie
Madame 1, nee le 11/01/2004 21:41:44, est morte le 11/01/2004 21:41:46.
Monsieur 2, ne le 11/01/2004 21:41:44, est marie depuis le 11/01/2004 21:41:46.
Monsieur 3, ne le 11/01/2004 21:41:44, est mort le 11/01/2004 21:41:46.
Monsieur 2, ne le 11/01/2004 21:41:44, est veuf depuis le 11/01/2004 21:41:48.
Monsieur 0, ne le 11/01/2004 21:41:44, est marie depuis le 11/01/2004 21:41:50.
Monsieur 0, ne le 11/01/2004 21:41:44, est divorce depuis le 11/01/2004 21:41:52.
Monsieur 2, ne le 11/01/2004 21:41:44, est mort le 11/01/2004 21:41:54.
Monsieur 0, ne le 11/01/2004 21:41:44, est mort le 11/01/2004 21:41:56.
        Situation a la date 11/01/2004 21:41:50
Monsieur 0, ne le 11/01/2004 21:41:44, est celibataire.
Madame 1, nee le 11/01/2004 21:41:44, est morte le 11/01/2004 21:41:46.
Monsieur 2, ne le 11/01/2004 21:41:44, est veuf depuis le 11/01/2004 21:41:48.
Monsieur 3, ne le 11/01/2004 21:41:44, est mort le 11/01/2004 21:41:46.
```

Question 4.1 (4 points) *Complétez (lignes 12-15, 25, 32-35, 41, 45-46 et 50-56) le code de la classe* `Personne`.

```
1   import java.util.*;
2   public class Personne extends Thread{
3       private static int numero = 0;
4       private static Random alea = new Random();
5
6       private int ident;
7       private boolean masculin;
8       private List vie = new ArrayList(); // pour gerer l'historique
9       private Situation situation;
10
11      public Personne(boolean garcon){
12
13
14
15
16      }
17
18      public int     getIdent()    {return ident;}
19      public boolean isMasculin()  {return masculin;}
20      public Date    getNaissance(){return ((Situation)vie.get(0)).getDebut();}
21      public Date    getDeces()    {return situation.getDeces();}
22      public String  toString()    {return situation.toString();}
23
24      public void run(){ // Remplace vieillir()
25
26              try {Thread.sleep(2000);}
27              catch(InterruptedException ie){}
28              finally {
29                  int a = alea.nextInt(4);
30                  try {
31                      switch (a) {
32                      case 0 :                              break;
33                      case 1 :                              break;
34                      case 2 :                              break;
35                      case 3 :
36                      }
37                      System.out.print(this);
38                  }
39                  catch(Exception e){}
40              }
41
42      }
43
44      public void changerSituation(Situation sit){
45
46
47      }
48
49      public Situation getSituation(Date evt){
50
51
52
53
54
55
56
```

```
        }
}
```

Question 4.2 (1 point) *Expliquez pourquoi, cette solution ne fonctionne pas si on l'implémente directement avec la méthode* `vieillir()`.