Interfaces Humain-Machine

I2 - Algorithmique et Programmation structurée

Julien Saunier, Alexandre Pauchet, Pierrick Tranouez

Plan

Rappel : Modularité

Objectifs de la modularité 1 / 1

Définition (selon Wikipedia)

En informatique, la programmation modulaire consiste dans le regroupement de fonctions, de méthodes et de traitement.

Objectifs

- Distribution du codage dans une équipe
- Possibilité de faire des tests unitaires
- Réutilisation de codes
- Réalisation de bibliothèques logicielles

Unités en Pascal 1 / 1

Définition générale

Le rôle d'une unité est de **regrouper** dans un même module l'implémentation d'un **ensemble de définitions de types, de fonctions et de procédures** d'un **même thème** dans un objectif de **réutilisabilité**.

- une unité est une bibliothèque logicielle,
- nécessite une documentation
- la distribution de base de freepascal contient des unités (la RTL)

Usage des unités

Syntaxe

pour utiliser le contenu d'une unité dans un autre programme, il faut l'"importer" par la commande `uses`.

Interface

Tous les éléments déclarés dans l'interface sont utilisables :

- Constantes, types personnalisés
- Fonctions, procédures

Utilisation

Deux types d'unités :

On peut

- créer ses propres unités [génie logiciel]
- utiliser des unités existantes (bibliothèques) :
  `https://www.freepascal.org/docs-html/rtl/index.html`
  - Unité system importée systématiquement
  - e.g. readln, writeln...
    `https://www.freepascal.org/docs-html/rtl/system/index.html`

Unités FreePascal

Quelques unités FreePascal

Liste et documentation

`http://www.freepascal.org/docs-html/rtl/index.html`

- `Math`
  - Fonctions mathématiques
- `Sysutils`
  - Instructions standards d'accès à des fonctions du système
- `Crt`
  - Gestion de l'interface textuelle par le clavier et l'écran
- `Keyboard`
  - Interactions directes avec les événements issus du clavier

Unité Math

Quelques fonctions et procédures

- `cos`, `arccos`, `sin`, ...
  - Fonctions trigonométrique
- `ceil`, `floor`, `roundTo`, `max`, ...
  - Arrondit ou compare des valeurs numériques
- `intPower`, `log2`, `logn`, ...
  - Fonctions mathématiques diverses
- ...

Unité Sysutils

Quelques fonctions et procédures

- `BoolToStr`, `StrToBool`, `IntToStr`, `StrToInt`, `FloatToStr`, `StrToFloat`,
  - Conversions de type
- `CreateDir`, `RemoveDir`, `RenameFile`, `FileExists`
  - Manipulation du système de fichier
- `Now`, `Date`, `Time`, `DateToStr`, `TimeToStr`
  - Manipulations de dates et d'heures
- ...

Exemple d'utilisation de Sysutils

Quelques fonctions et procédures

Écrire un programme qui donne la date et l'heure actuelles, puis demande un nombre de dossier à créer (et les crée).

IHM

IHM

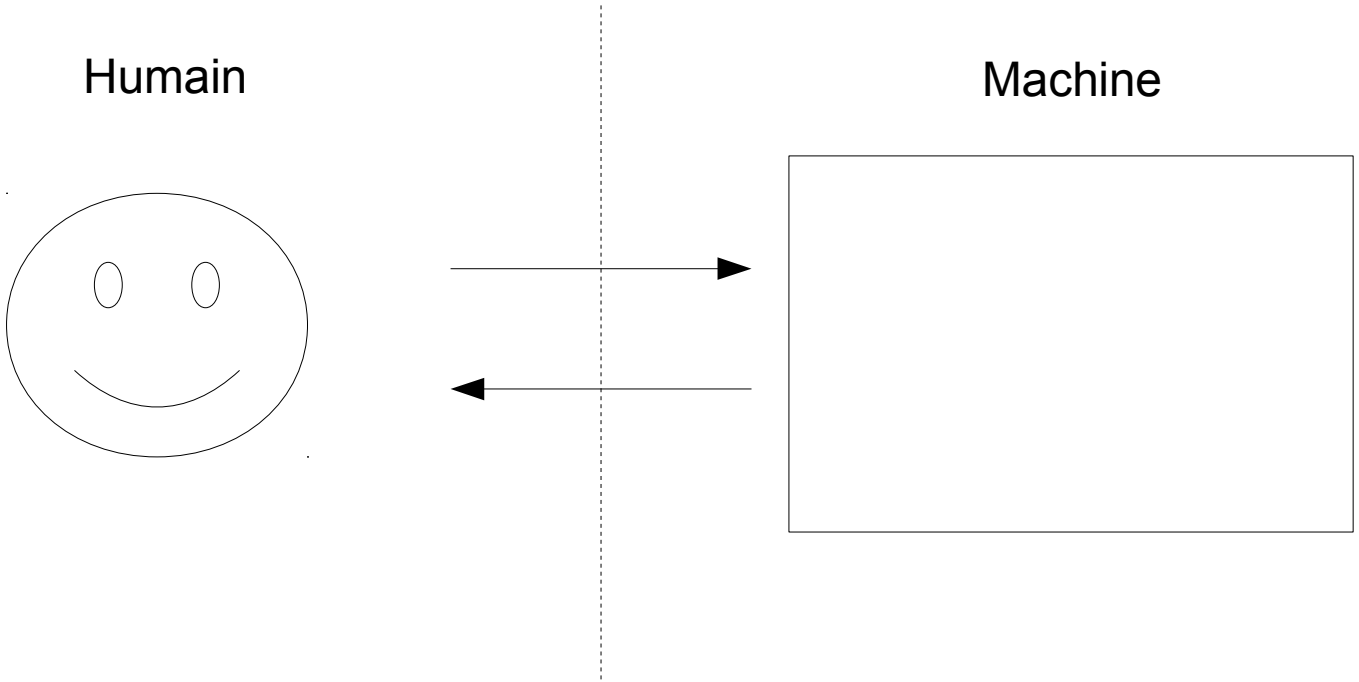
Définition générale

Les interactions Humain-machines (IHM) définissent les moyens et outils mis en œuvre afin qu'un humain puisse contrôler et communiquer avec une machine.

Objectifs

- ergonomie
- efficacité
- adapté au contexte

Boucle



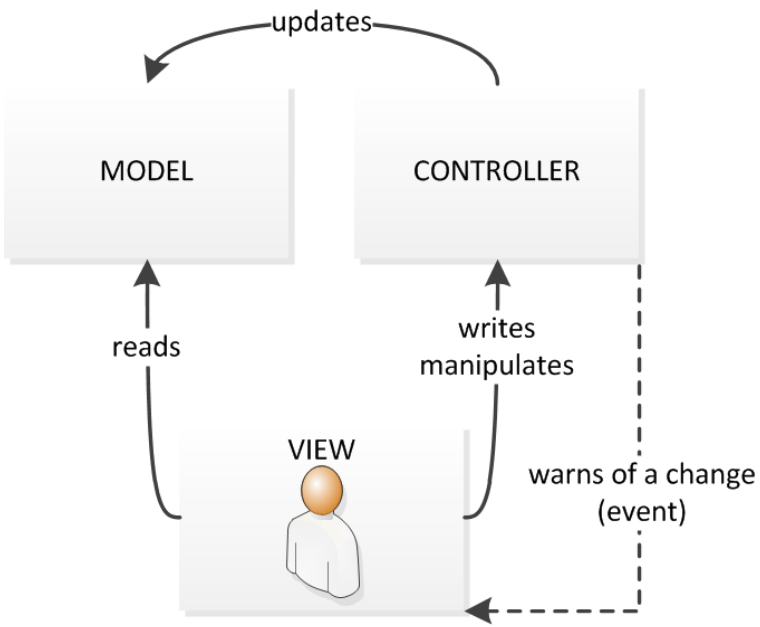
- Gestion des entrées (captation de l'utilisateur)
- Gestion des sorties (réponse du programme)

Séparer la logique du programme de l'IHM

Dans le terminal

- Indépendance entre logique du programme et présentation
  - Un logiciel de gestion d'agenda ne s'affiche pas de la même façon sur mobile ou fixe, les modes d'interactions sont différents
  - Mais la gestion des événements est la même
- A prévoir dès la conception globale
  - Le coeur du programme ne change pas
  - Seules le choix des procédures d'interaction change

Exemple : Modèle-Vue-Contrôleur



- Modèle : gestion des données ainsi que de la logique en rapport avec les données : validation, lecture et enregistrement
- Vue : partie visible d'une interface graphique, contenant des éléments visuels ainsi que la logique nécessaire pour afficher les données provenant du modèle.
- Contrôleur :traitement des actions de l'utilisateur, modification des données du modèle et de la vue

Indépendance Données - Visualisation

Exemple

- Stockage des informations d'un labyrinthe
- Comment représenter les informations dans le programme ?
- Comment les afficher ?

Unités dédiés à l'IHM en Pascal

Dans le terminal

- Gestion des entrées : Keyboard
- Gestion des sorties : Crt

Peut être remplacé par d'autres bibliothèques

- SDL

Unité Crt : par rapport à write/writeln

Changement de logique

- Avec writeln : affichage séquentiel des éléments l'un après l'autre
- Avec crt : possibilité de se déplacer dans le terminal et ré-écrire des informations
  - Attention : l'ordre d'exécution est toujours séquentiel

Unité Crt

Quelques fonctions et procédures

- `ClrScr`, `ClrEol`
  - Efface l'écran ou la fin d'une ligne depuis un curseur
- `GotoXY`, `WhereX`, `WhereY`
  - Positionne ou renvoie les coordonnées du curseur
- `TextColor`, `TextBackground`
  - Change la couleur du texte ou du fond
- ...

Exemple d'utilisation de Crt

Quelques fonctions et procédures

Écrire un programme qui affiche une grille avec les numéros des lignes et colonnes inscrits en couleur. Des coordonnées X, Y peuvent être saisies en haut de la grille et une croix apparaît aux coordonnées saisies. quand une valeur négative est entrée le programme se termine.

Unité Keyboard : par rapport à read/readln

Changement de logique

- Avec readln : récupération des informations tapés au clavier
- Avec keyboard : interception des événements qui ne sont plus nécéssairement affichés

Unité Keyboard

Quelques fonctions et procédures

- `InitKeyboard`, `DoneKeyboard`
  - Démarre ou termine le mode d'accès direct au clavier
- `GetKeyEvent`, `TranslateKeyEvent`
  - Récupère ou traduit un événement clavier
- `GetKeyEventChar`, `GetKeyEventCode`, `GetKeyEventFlags`
  - Donne des indications sur un événement
- ...

Exemple d'utilisation de Keyboard

Quelques fonctions et procédures

Écrire un programme qui demande de saisir un mot de passe. Le texte saisi ne doit pas être affiché à l'écran,chaque caractère étant remplacé par une étoile (*). Si le mot de passe entré est "INSA" le programme affiche "ACCES ACCORDE" sinon il affiche "ACCES REFUSE".

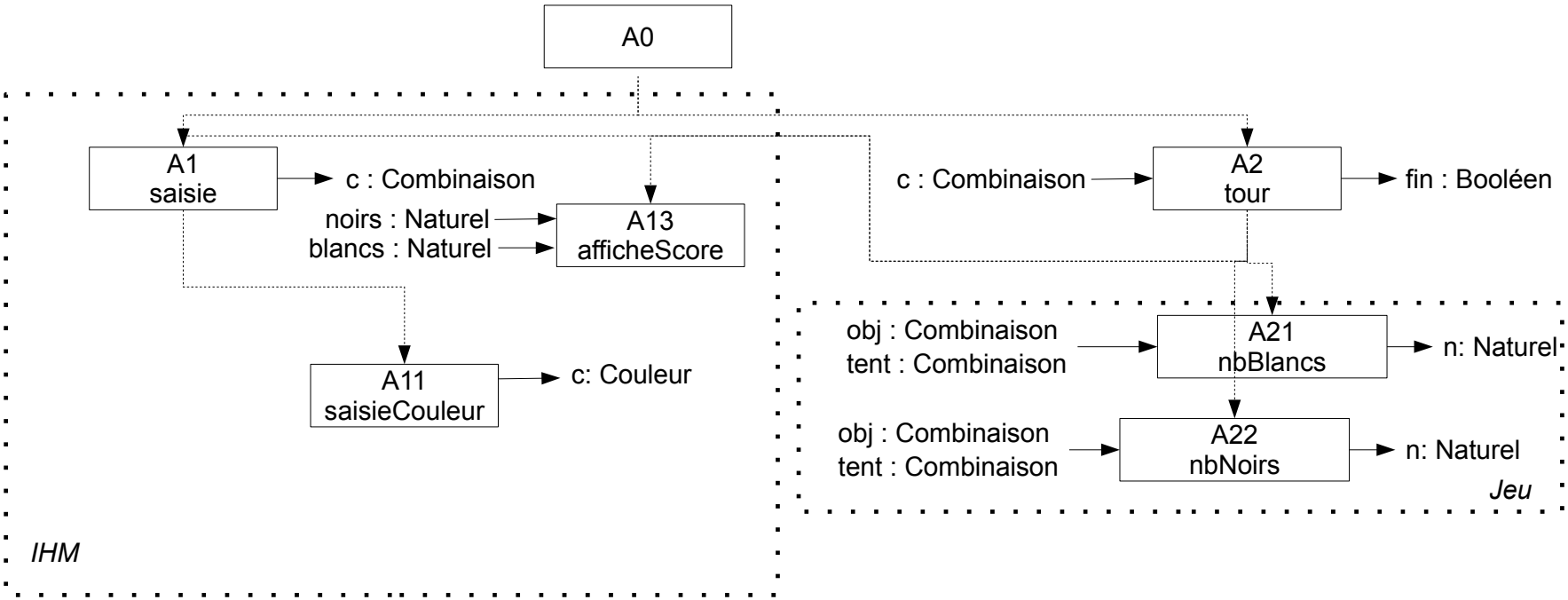
Exemple :Jeu du mastermind 1 / 11

Quelques fonctions et procédures

Écrire une nouvelle version du jeu du mastermind. Cette version s'appuiera sur des unités pour :

1. séparer le code source en une unité pour la gestion de l'interface homme-machine, une pour le calcul des pions blancs et noirs des règles du jeu et une pour le programme principal gérant l'ensemble du jeu ;
2. s'appuyer sur les unités Crt et Keyboard pour un jeu plus convivial.

Exemple :Jeu du mastermind 2 / 11



Exemple :Jeu du mastermind 3 / 11

mastermind.pas

```
program mastermind;

uses Crt, mastermindJeu, mastermindIhm;

{Cette procedure gere un tour de jeu.
"objectif" correspond a la combinaison a trouver. "victoire" vaut Vrai si le tour est gagnant
procedure tour(objectif : Combinaison; var victoire : boolean);
var tentative    : Combinaison;
   blancs, noirs :  integer;
begin
   saisie(tentative);
   noirs := nbNoirs(objectif, tentative);
   blancs := nbBlancs(objectif, tentative);
   afficheScore(noirs, blancs);
   writeln();
   if (noirs = 4) then
      victoire := true
   else
      victoire := false;
end;
```

Exemple :Jeu du mastermind 4 / 11

mastermind.pas

```
{Programme principal}
var obj   : Combinaison;
   round : integer;
   gagne :  boolean;
begin
   {Le joueur 1 saisit la combinaison a trouver}
   writeln('Joueur 1, entrez la combinaison a deviner');
   saisie(obj);
   gagne := false;
   round := 1;
   ClrScr();
   {Le joueur 2 propose des tentatives jusqu'a 10 propositions maximum ou qu'il trouve la bo
   writeln('Joueur 2, a vous de deviner la combinaison');
   repeat
      writeln('Tentative no', round);
      tour(obj, gagne);
      round := round + 1;
   until gagne or (round > 10);
   if gagne then
      writeln('Bravo vous avez trouve la bonne combinaison')
   else
      writeln('Game over, vous avez perdu...');
end.
```

Exemple :Jeu du mastermind 5 / 11

mastermindJeu.pas

```
unit mastermindJeu;

interface

Type Couleur = (jaune, orange, mauve, rouge, vert, bleu);
Type Combinaison =  array[1..4] of Couleur;

function nbNoirs(objectif, tentative :  Combinaison) : integer;
function nbBlancs(objectif, tentative :  Combinaison) : integer;

implementation

{Cette fonction renvoie le nombre de pions noirs a afficher en fonction d'une proposition. "
function nbNoirs(objectif, tentative :  Combinaison) : integer;
var i, resultat :  integer;
begin
   resultat := 0;
   for i := 1 to 4 do
      if (objectif[i] = tentative[i]) then resultat := resultat + 1;
   nbNoirs := resultat;
end;
```

Exemple :Jeu du mastermind 6 / 11

mastermindJeu.pas

```
{Cette fonction renvoie le nombre de pions blancs a afficher en fonction d'une proposition.
function nbBlancs(objectif, tentative :  Combinaison) : integer;
var i, resultat : integer;
   col          : Couleur;
   nbCouleurs   :  array[jaune..bleu] of integer;
begin
   resultat := 0;
   for col := jaune to bleu do
      nbCouleurs[col] := 0;
   for i := 1 to 4 do
      if (objectif[i] <> tentative[i]) then
         nbCouleurs[objectif[i]] := nbCouleurs[objectif[i]] +1;
   for i := 1 to 4 do
      if (objectif[i] <> tentative[i]) and (nbCouleurs[tentative[i]] > 0) then
      begin
         resultat := resultat + 1;
         nbCouleurs[tentative[i]] := nbCouleurs[tentative[i]] - 1;
      end;
   nbBlancs := resultat;
end;

end.
```

Exemple :Jeu du mastermind 7 / 11

mastermindIhm.pas

```
unit mastermindIhm;

interface

uses Crt, Keyboard, mastermindJeu;

procedure saisie(var tab :  Combinaison);
procedure afficheScore(pn, pb : Integer);

implementation

{Cette procedure locale affiche en couleur un element d'une combinaison}
procedure afficheCouleur(c : Couleur);
begin
   case c of
     jaune :TextColor(Yellow);
     orange:  TextColor(LightRed);
     mauve :  TextColor(Magenta);
     rouge: TextColor(Red);
     vert:  TextColor(Green);
     bleu:  TextColor(Blue);
   end; { case }
   write(c);
   TextColor(LightGray);
end;
```

Exemple :Jeu du mastermind 8 / 11

mastermindIhm.pas

```
{Cette procedure affiche en couleur le nombre de pions noirs et de pions blancs d'une propos
procedure afficheScore(pn, pb : Integer);
begin
   TextColor(LightGray);
   write('Score : ');
   TextColor(Brown);
   write(pn, ' pions noirs ');
   TextColor(White);
   write(pb, ' pions blancs');
   TextColor(LightGray);
end;
```

Exemple :Jeu du mastermind 9 / 11

mastermindIhm.pas

```
{Cette procedure locale sert a saisir une couleur par une seule touche et permet d'annuler l
procedure saisieCouleur(var c : Couleur; var retour : Boolean);
var key    : TKeyEvent;
   ok      :  Boolean;
begin
   retour := false;
   repeat
      ok := true;
      key := GetKeyEvent();
      key := TranslateKeyEvent(key);
      case GetKeyEventChar(key) of
        'b' : c := bleu;
        'v' : c := vert;
        'o' : c := orange;
        'j' : c := jaune;
        'r' : c := rouge;
        'm' : c := mauve;
        'a' : retour := true;
      else ok := false;
      end;
   until (retour = true) or (ok = true);
end;
```

Exemple :Jeu du mastermind 10 / 11

mastermindIhm.pas

```
{Cette procedure demande a l'utilisateur de saisir une combinaison de 4 couleurs qui est aff
procedure saisie(var tab :  Combinaison);
var i, j : integer; annul : Boolean; c : Couleur;
begin
   InitKeyBoard();
   i := 1;
   writeln('Saisissez une combinaison de 4 couleurs');
   repeat
      saisieCouleur(c,annul);
      if (annul = true) then
         begin
            if (i > 1) then
            begin
               GotoXY(1,WhereY());
               ClrEol();
               i := i -1;
               for j:=1 to i-1 do
               begin
                  afficheCouleur(tab[j]);
                  write(' ');
               end
            end
         end
```

Exemple :Jeu du mastermind 11 / 11

mastermindIhm.pas

```
      else
         begin
            tab[i] := c;
            afficheCouleur(c);
            write('␣');
            i := i + 1;
         end;
   until i > 4;
   DoneKeyBoard();
end;

end.
```