UNIVERSITÉ LAVAL

PlanIFTicateur

Travail pratique 3

GLO-2004 – Génie Logiciel Orienté Objet – Hiver 2015

Travail présenté à
M. Jonathan Gaudreault

Chayer, Philippe	Philippe.chayer.1@ulaval.ca IFT	PHCHA47
Khouma, Abdou	abdou.khouma.1@ulaval.ca GIF	ABKHO9
Gadoury, Gabriel	Gabriel.gadoury.1@ulaval.ca IFT	GAGAD1
Yeo, Clotioloman	Clotioloman.yeo.1@ulaval.ca GLO	CLYEO1

Table des matières

Introduction

La construction des horaires de session consiste en une tâche difficile pour la direction, peu importe le département de l'université. À l'aide du logiciel que nous développons, cette tâche deviendra visuelle et interactive, c'est-à-dire plus facile à réaliser. Ainsi, la direction (ou responsable de la création de l'horaire) économisera beaucoup de temps donc, par le fait même, diminuera significativement le coût associé à la production d'un horaire optimal.

Le rapport ci-présent consiste à présenter le modèle de conception mis à jour et l'architecture logicielle de PlanIFTicateur, le logiciel de création d'horaire de session.

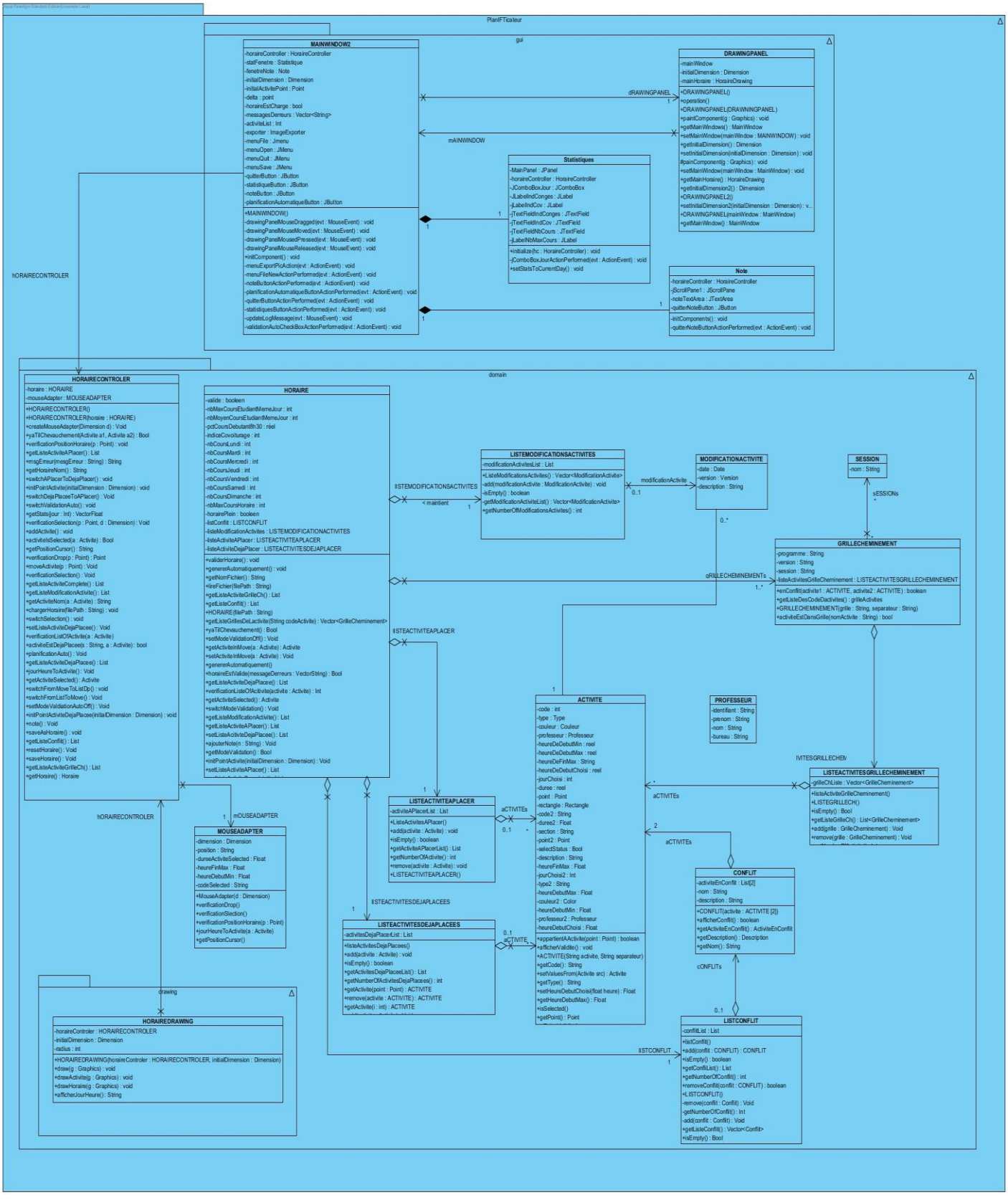
La première section montre un diagramme de classe. Il s'agit de l'architecture qui est utilisée afin d'implanter notre solution qui, soit dit en passant, sera sous le langage Java. Ensuite, une section est consacrée à illustrer les diagrammes d'états de certains événements de notre application. Suivra les diagrammes de séquence qui permettront de mieux visualiser les particularités fonctionnelles de PlanIFTicateur. Finalement, une dernière section illustrera la gestion du projet mise à jour ainsi que quelques éléments fondamentaux du rapport précédent.

Diagramme des classes de conception

Le diagramme qui suit représente le corps des classes utilisées par notre application. Notez qu'il y a beaucoup de changements qui ont été apportés par rapport au diagramme du rapport précédent. Tout d'abord, nous avons ajouté plusieurs méthodes et attributs dans les différentes classes, voir toutes. Certains attributs ont ajoutés suite à plusieurs constations lors de l'implémentation du code. Nous avons aussi ajouté une classe **MouseAdapter** qui suit le modèle d'un adaptateur permettant de faire certaines vérifications au niveau de la position de la souris pour le mouvement **Drag-and-Drop** au niveau de l'horaire. Notre contrôleur a également subi plusieurs modifications. Le contrôleur, nommé ici **HoraireController**, est la seule classe qui échange avec l'interface. Tout élément de l'interface qui souhaite accéder à une fonction ou un événement doit passer par ce contrôleur qui s'assurera, par la suite, de faire la relation vers le code approprié. Le même phénomène se produit à l'inverse, c'est-à-dire lorsque les éléments du domaine, qui contient le code fonctionnel, désirent interagir avec l'interface utilisateur.

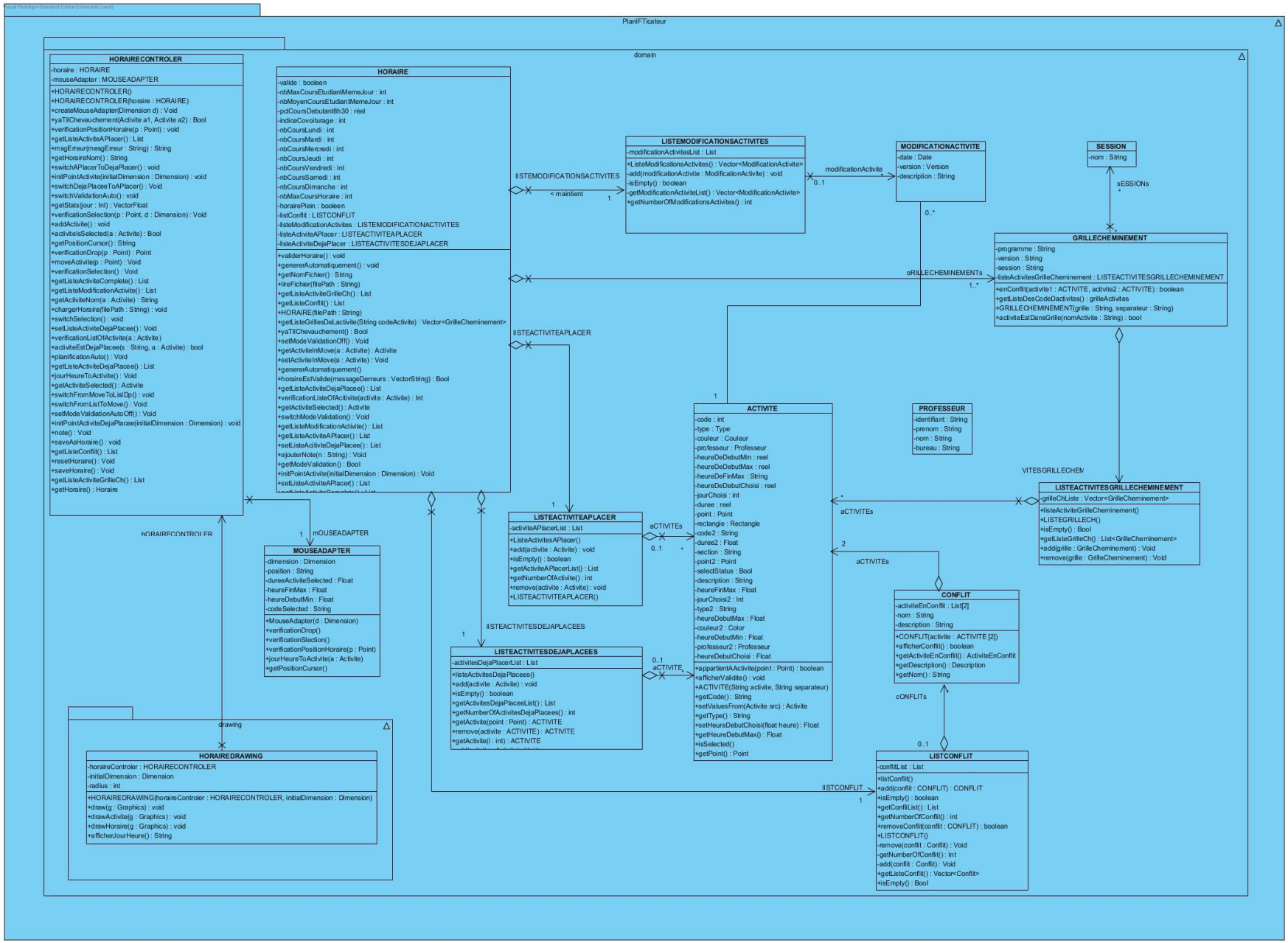
Au niveau de l'interface (**GUI**), vous observerez l'ajout d'une fenêtre/classe **Note** et d'une fenêtre/classe **Statistiques**. Ces deux fenêtres gèrent des actions sur une interface différente, cependant elles restent liées à la fenêtre principale (**MainWindow**) et n'ont aucune signification sans cette dernière.

(Schéma sur page suivante.)


PlaniFTicateur
gui
MAINWINDOW2
-horaireController : HoraireController
-statFenetre : Statistique
-fenetreNote : Note
-initialDimension : Dimension
-initialActivitePoint : Point
-delta : point
-horaireEstCharge : bool
-messagesDerreurs : Vector<String>
-activiteList : Int
-exporter : ImageExporter
-menuFile : Jmenu
-menuOpen : JMenu
-menuQuit : JMenu
-menuSave : JMenu
-quitterButton : JButton
-statistiqueButton : JButton
-noteButton : JButton
-planificationAutomatiqueButton : JButton
+MAINWINDOW()
-drawingPanelMouseDragged(evt : MouseEvent) : void
-drawingPanelMouseMoved(evt : MouseEvent) : void
-drawingPanelMousedPressed(evt : MouseEvent) : void
-drawingPanelMouseReleased(evt : MouseEvent) : void
+initComponent() : void
-menuExportPicActionje(vt : ActionEvent) : void
-menuFileNewActionPerformed(evt : ActionEvent) : void
-noteButtonActionPerformed(evt : ActionEvent) : void
-planificationAutomatiqueButtonActionPerformed(evt : ActionEvent) : void
-quitterButtonActionPerformed(evt : ActionEvent) : void
-statistiquesButtonActionPerformed(evt : ActionEvent) : void
-updateLogMessage(evt : MouseEvent) : void
-validationAutoCheckBoxActionPerformed(evt : ActionEvent) : void
dRAWINGPANEL
mAINWINDOW
DRAWINGPANEL
-mainWindow
-initialDimension : Dimension
-mainHoraire : HoraireDrawing
+DRAWINGPANEL()
+operation()
+DRAWINGPANEL(DRAWNINGPANEL)
+paintComponent(g : Graphics) : void
+getMainWindows() : MainWindow
+setMainWindow(mainWindow : MAINWINDOW) : void
+getInitialDimension() : Dimension
+setInitialDimension(initialDimension : Dimension) : void
#painComponent(g : Graphics) : void
+setMainWindow(mainWindow : MainWindow) : void
+getMainHoraire() : HoraireDrawing
+getInitialDimension2() : Dimension
+DRAWINGPANEL2()
+setInitialDimension2(initialDimension : Dimension) : v...
+DRAWINGPANEL(mainWindow : MainWindow)
+getMainWindow() : MainWindow
Statistiques
-MainPanel : JPanel
-horaireController : HoraireController
-jComboBoxJour : JComboBox
-JLabelIndConges : JLabel
-jLabelIndCov : JLabel
-jTextFieldIndConges : JTextField
-jTextFieldIndCov : JTextField
-jTextFieldNbCours : JTextField
-jLabelNbMaxCours : JLabel
+initialize(hc : HoraireController) : void
-jComboBoxJourActionPerformed(evt : ActionEvent) : void
+setStatsToCurrentDay() : void
Note
-horaireController : HoraireController
-jScrollPane1 : JScrollPane
-noteTextArea : JTextArea
-quitterNoteButton : JButton
-initComponents() : void
-quitterNoteButtonActionPerformed(evt : ActionEvent) : void
hORAIRECONTROLER
domain
HORAIRECONTROLER
-horaire : HORAIRE
-mouseAdapter : MOUSEADAPTER
+HORAIRECONTROLER()
+HORAIRECONTROLER(horaire : HORAIRE)
+createMouseAdapter(Dimension d) : Void
+yaTilChevauchement(Activite a1 , Activite a2) : Bool
+verificationPositionHoraire(p : Point) : void
+getListeActiviteAPlacer() : List
+msgErreur(messgErreur : String) : String
+getHoraireNom() : String
+switchAPlacerToDejaPlacer() : void
+initPointActivite(initialDimension : Dimension) : void
+switchDejaPlaceToAPlacer() : Void
+switchValidationAuto() : void
+getStats(jour : Int) : VectorFloat
+verificationSelection(p : Point, d : Dimension) : Void
+addActivite() : void
+activiteIsSelected(a : Activite) : Bool
+getPositionCursor() : String
+verificationDrop(p : Point) : Point
+moveActivite(p : Point) : Void
+verificationSelection() : Void
+getListeActiviteComplete() : List
+getListeModificationActivite() : List
+getActiviteNom(a : Activite) : String
+chargerHoraire(filePath : String) : void
+switchSelection() : void
+setListeActiviteDejaPlacee() : Void
+verificationListOfActivite(a : Activite)
+activiteEstDejaPlacee(s : String, a : Activite) : bool
+planificationAuto() : Void
+getListeActiviteDejaPlacee() : List
+jourHeureToActivite() : Void
+getActiviteSelected() : Activite
+switchFromMoveToListOp() : void
+switchFromListToMove() : Void
+setModeValidationAutoOff() : Void
+initPointActiviteDejaPlacee(initialDimension : Dimension) : void
+note() : Void
+saveAsHoraire() : void
+getListeConflit() : List
+resetHoraire() : Void
+saveHoraire() : Void
+getListeActiviteGrilleCh() : List
+getHoraire() : Horaire
HORAIRE
-valide : booleen
-nbMaxCoursEtudiantMemeJour : int
-nbMoyenCoursEtudiantMemeJour : int
-pctCoursDebutant8h30 : réel
-indiceCovoiturage : int
-nbCoursLundi : int
-nbCoursMardi : int
-nbCoursMercredi : int
-nbCoursJeudi : int
-nbCoursVendredi : int
-nbCoursSamedi : int
-nbCoursDimanche : int
-nbMaxCoursHoraire : int
-horairePlein : booleen
-listConflit : LISTCONFLIT
-listeModificationActivites : LISTEMODIFICATIONACTIVITES
-listeActiviteAPlacer : LISTEACTIVITEAPLACER
-listeActiviteDejaPlacer : LISTEACTIVITESDEJAPLACER
+validerHoraire() : void
+genererAutomatiquement() : void
+getNomFichier() : String
+lireFichier(filePath : String)
+getListeActiviteGrilleCh() : List
+getListeConflit() : List
+HORAIRE(filePath : String)
+getListeGrillesDeLactivite(String codeActivite) : Vector<GrilleCheminement>
+yaTilChevauchement() : Bool
+setModeValidationOff() : Void
+getActiviteInMove(a : Activite) : Activite
+setActiviteInMove(a : Activite) : Void
+genererAutomatiquement()
+horaireEstValide(messageDerreurs : VectorString) : Bool
+getListeActiviteDejaPlacee() : List
+verificationListeOfActivite(activite : Activite) : Int
+getActiviteSelected() : Activite
+switchModeValidation() : Void
+getListeModificationActivite() : List
+getListeActiviteAPlacer() : List
+setListeActiviteDejaPlacee() : List
+ajouterNote(n : String) : Void
+getModeValidation() : Bool
+initPointActivite(initialDimension : Dimension) : Void
+setListeActiviteAPlacer() : List
lISTEMODIFICATIONSACTIVITES
< maintient
LISTEMODIFICATIONSACTIVITES
-modificationActivitesList : List
+ListeModificationsActivites() : Vector<ModificationActivite>
-add(modificationActivite : ModificationActivite) : void
-isEmpty() : boolean
-getModificationActiviteList() : Vector<ModificationActivite>
+getNumberOfModificationsActivites() : int
modificationActivite
MODIFICATIONACTIVITE
-date : Date
-version : Version
-description : String
SESSION
-nom : String
sESSIONs
gRILLECHEMINEMENTs
GRILLECHEMINEMENT
-programme : String
-version : String
-session : String
-listeActivitesGrilleCheminement : LISTEACTIVITESGRILLECHEMINEMENT
+enConflit(activite1 : ACTIVITE, activite2 : ACTIVITE) : boolean
+getListeDesCodeDactivites() : grilleActivites
+GRILLECHEMINEMENT(grille : String, separateur : String)
+activiteEstDansGrille(nomActivite : String) : bool
lISTEACTIVITEAPLACER
LISTEACTIVITEAPLACER
-activiteAPlacerList : List
+ListeActivitesAPlacer()
+add(activite : Activite) : void
+isEmpty() : boolean
+getActiviteAPlacerList() : List
+getNumberOfActivite() : int
+remove(activite : Activite) : void
+LISTEACTIVITEAPLACER()
aCTIVITEs
ACTIVITE
-code : int
-type : Type
-couleur : Couleur
-professeur : Professeur
-heureDeDebutMin : reel
-heureDeDebutMax : reel
-heureDeFinMax : String
-heureDeDebutChoisi : reel
-jourChoisi : int
-duree : reel
-point : Point
-rectangle : Rectangle
-code2 : String
-duree2 : Float
-section : String
-point2 : Point
-selectStatus : Bool
-description : String
-heureFinMax : Float
-jourChoisi2 : Int
-type2 : String
-heureDebutMax : Float
-couleur2 : Color
-heureDebutMin : Float
-professeur2 : Professeur
-heureDebutChoisi : Float
+appartientAActivite(point : Point) : boolean
+afficherValidite() : void
+ACTIVITE(String activite, String separateur)
+getCode() : String
+setValuesFrom(Activite src) : Activite
+getType() : String
+setHeureDebutChoisi(float heure) : Float
+getHeureDebutMax() : Float
+isSelected()
+getPoint() : Point
PROFESSEUR
-identifiant : String
-prenom : String
-nom : String
-bureau : String
IVITESGRILLECHEM
LISTEACTIVITESGRILLECHEMINEMENT
-grilleChListe : Vector<GrilleCheminement>
+listeActiviteGrilleCheminement()
+LISTEGRILLECH()
+isEmpty() : Bool
+getListeGrilleCh() : List<GrilleCheminement>
+add(grille : GrilleCheminement) : Void
+remove(grille : GrilleCheminement) : Void
aCTIVITEs
2
aCTIVITEs
CONFLIT
-activiteEnConflit : List[2]
-nom : String
-description : String
+CONFLIT(activite : ACTIVITE [2])
+afficherConflit() : boolean
+getActiviteEnConflit() : ActiviteEnConflit
+getDescription() : Description
+getNom() : String
cONFLITs
0..1
LISTCONFLIT
-conflitList : List
+listConflit()
+add(confit : CONFLIT) : CONFLIT
+isEmpty() : boolean
+getConflitList() : List
+getNumberOfConflit() : int
+removeConflit(confit : CONFLIT) : boolean
+LISTCONFLIT()
-remove(confit : Confit) : Void
-getNumberOfConfit() : Int
-add(confit : Confit) : Void
+getListeConfit() : Vector<Confit>
+isEmpty() : Bool
lISTCONFLIT
hORAIRECONTROLER
mOUSEADAPTER
MOUSEADAPTER
-dimension : Dimension
-position : String
-dureeActiviteSelected : Float
-heureFinMax : Float
-heureDebutMin : Float
-codeSelected : String
+MouseAdapter(d : Dimension)
+verificationDrop()
+verificationSlection()
+verificationPositionHoraire(p : Point)
+jourHeureToActivite(a : Activite)
+getPositionCursor()
lISTEACTIVITESDEJAPLACEES
LISTEACTIVITESDEJAPLACEES
-activitesDejaPlacerList : List
+listeActivitesDejaPlacees()
+add(activite : Activite) : void
+isEmpty() : boolean
+getActivitesDejaPlaceeList() : List
+getNumberOfActivitesDejaPlacees() : int
+getActivite(point : Point) : ACTIVITE
+remove(activite : ACTIVITE) : ACTIVITE
+getActivite(i : int) : ACTIVITE
drawing
HORAIREDRAWING
-horaireControler : HORAIRECONTROLER
-initialDimension : Dimension
-radius : int
+HORAIREDRAWING(horaireControler : HORAIRECONTROLER, initialDimension : Dimension)
+draw(g : Graphics) : void
+drawActivite(g : Graphics) : void
+drawHoraire(g : Graphics) : void
+afficherJourHeure() : String

******Étant donné la grande taille du schéma, nous avons séparé le schéma du domaine et de l'interface pour permettre une lecture plus facile des différentes parties.***

(Il est aussi possible de le voir plus facilement à l'aide de visual paradigm. Voir fichier joint.)


PlaniFTicateur
domain
HORAIRECONTROLER
-horaire : HORAIRE
-mouseAdapter : MOUSEADAPTER
+HORAIRECONTROLER()
+HORAIRECONTROLER(horaire : HORAIRE)
+createMouseAdapter(Dimension d) : Void
+yaTilChevauchement(Activite a1, Activite a2) : Bool
+verificationPositionHoraire(p : Point) : void
+getListeActiviteAPlacer() : List
+msgErreur(mesgErreur : String) : String
+getHoraireNom() : String
+switchAPlacerToDejaPlacer() : void
+initPointActivite(initialDimension : Dimension) : void
+switchDejaPlaceToAPlacer() : Void
+switchValidationAuto() : void
+getStats(jour : Int) : VectorFloat
+verificationSelection(p : Point, d : Dimension) : Void
+addActivite() : void
+activiteIsSelected(a : Activite) : Bool
+getPositionCursor() : String
+verificationDrop(p : Point) : Point
+moveActivite(p : Point) : Void
+verificationSelection() : Void
+getListeActiviteComplete() : List
+getListeModificationActivite() : List
+getActiviteNom(a : Activite) : String
+chargerHoraire(filePath : String) : void
+switchSelection() : void
+setListeActiviteDejaPlacee() : Void
+verificationListOfActivite(a : Activite)
+activiteEstDejaPlacee(s : String, a : Activite) : bool
+planificationAuto() : Void
+getListeActiviteDejaPlacee() : List
+jourHeureToActivite() : Void
+getActiviteSelected() : Activite
+switchFromMoveToListDp() : void
+switchFromListToMove() : Void
+setModeValidationAutoOff() : Void
+initPointActiviteDejaPlacee(initialDimension : Dimension) : void
+note() : Void
+saveAsHoraire() : void
+getListeConflit() : List
+resetHoraire() : Void
+saveHoraire() : Void
+getListeActiviteGrilleCh() : List
+getHoraire() : Horaire
HORAIRE
-valide : booleen
-nbMaxCoursEtudiantMemeJour : int
-nbMoyenCoursEtudiantMemeJour : int
-pctCoursDebutant8h30 : réel
-indiceCovoiturage : int
-nbCoursLundi : int
-nbCoursMardi : int
-nbCoursMercredi : int
-nbCoursJeudi : int
-nbCoursVendredi : int
-nbCoursSamedi : int
-nbCoursDimanche : int
-nbMaxCoursHoraire : int
-horairePlein : booleen
-listConflit : LISTCONFLIT
-listeModificationActivites : LISTEMODIFICATIONACTIVITES
-listeActiviteAPlacer : LISTEACTIVITEAPLACER
-listeActiviteDejaPlacer : LISTEACTIVITESDEJAPLACER
+validerHoraire() : void
+genererAutomatiquement() : void
+getNomFichier() : String
+lireFichier(filePath : String)
+getListeActiviteGrilleCh() : List
+getListeConflit() : List
+HORAIRE(filePath : String)
+getListeGrillesDeLactivite(String codeActivite) : Vector<GrilleCheminement>
+yaTilChevauchement() : Bool
+setModeValidationOff() : Void
+getActiviteInMove(a : Activite) : Activite
+setActiviteInMove(a : Activite) : Void
+genererAutomatiquement()
+horaireEstValide(messageDerreurs : VectorString) : Bool
+getListeActiviteDejaPlacee() : List
+verificationListeOfActivite(activite : Activite) : Int
+getActiviteSelected() : Activite
+switchModeValidation() : Void
+getListeModificationActivite() : List
+getListeActiviteAPlacer() : List
+setListeActiviteDejaPlacee() : List
+ajouterNote(n : String) : Void
+getModeValidation() : Bool
+initPointActivite(initialDimension : Dimension) : Void
+setListeActiviteAPlacer() : List
lISTEMODIFICATIONSACTIVITES
< maintient
1
LISTEMODIFICATIONSACTIVITES
-modificationActivitesList : List
+ListeModificationsActivites() : Vector<ModificationActivite>
-add(modificationActivite : ModificationActivite) : void
-isEmpty() : boolean
-getModificationActiviteList() : Vector<ModificationActivite>
+getNumberOfModificationsActivites() : int
modificationActivite
0..1
*
MODIFICATIONACTIVITE
-date : Date
-version : Version
-description : String
0..*
SESSION
-nom : String
sESSIONs
*
GRILLECHEMINEMENT
-programme : String
-version : String
-session : String
-listeActivitesGrilleCheminement : LISTEACTIVITESGRILLECHEMINEMENT
+enConflit(activite1 : ACTIVITE, activite2 : ACTIVITE) : boolean
+getListeDesCodeDactivites() : grilleActivites
+GRILLECHEMINEMENT(grille : String, separateur : String)
+activiteEstDansGrille(nomActivite : String) : bool
aGRILLECHEMINEMENTs
1..*
lISTEACTIVITEAPLACER
-activiteAPlacerList : List
+ListeActivitesAPlacer()
+add(activite : Activite) : void
+isEmpty() : boolean
+getActiviteAPlacerList() : List
+getNumberOfActivite() : int
+remove(activite : Activite) : void
+LISTEACTIVITEAPLACER()
aCTIVITEs
0..1
*
lISTEACTIVITESDEJAPLACEES
-activitesDejaPlacerList : List
+listeActivitesDejaPlacees()
+add(activite : Activite) : void
+isEmpty() : boolean
+getActivitesDejaPlaceeList() : List
+getNumberOfActivitesDejaPlacees() : int
+getActivite(point : Point) : ACTIVITE
+remove(activite : ACTIVITE) : ACTIVITE
+getActivite(i : int) : ACTIVITE
0..1
aCTIVITE
*
ACTIVITE
-code : int
-type : Type
-couleur : Couleur
-professeur : Professeur
-heureDeDebutMin : reel
-heureDeDebutMax : reel
-heureDeFinMax : String
-heureDeDebutChoisi : reel
-jourChoisi : int
-duree : reel
-point : Point
-rectangle : Rectangle
-code2 : String
-duree2 : Float
-section : String
-point2 : Point
-selectStatus : Bool
-description : String
-heureFinMax : Float
-jourChois2 : Int
-type2 : String
-heureDebutMax : Float
-couleur2 : Color
-heureDebutMin : Float
-professeur2 : Professeur
-heureDebutChoisi : Float
+appartientAActivite(point : Point) : boolean
+afficherValidite() : void
+ACTIVITE(String activite, String separateur)
+getCode() : String
+setValuesFrom(Activite src) : Activite
+getType() : String
+setHeureDebutChoisi(float heure) : Float
+getHeureDebutMax() : Float
+isSelected()
+getPoint() : Point
1
PROFESSEUR
-identifiant : String
-prenom : String
-nom : String
-bureau : String
VITESGRILLECHEM
LISTEACTIVITESGRILLECHEMINEMENT
-grilleChListe : Vector<GrilleCheminement>
+listeActiviteGrilleCheminement()
+LISTEGRILLECH()
+isEmpty() : Bool
+getListeGrilleCh() : List<GrilleCheminement>
+add(grille : GrilleCheminement) : Void
+remove(grille : GrilleCheminement) : Void
*
aCTIVITEs
2
aCTIVITEs
CONFLIT
-activiteEnConflit : List[2]
-nom : String
-description : String
+CONFLIT(activite : ACTIVITE [2])
+afficherConflit() : boolean
+getActiviteEnConflit() : ActiviteEnConflit
+getDescription() : Description
+getNom() : String
cONFLITs
0..1
LISTCONFLIT
-conflitList : List
+listConflit()
+add(conflit : CONFLIT) : CONFLIT
+isEmpty() : boolean
+getConfliList() : List
+getNumberOfConflit() : int
+removeConflit(conflit : CONFLIT) : boolean
+LISTCONFLIT()
-remove(conflit : Conflit) : Void
-getNumberOfConflit() : Int
-add(conflit : Conflit) : Void
+getListeConflit() : Vector<Conflit>
+isEmpty() : Bool
lISTCONFLIT
1
hORAIRECONTROLER
1
mOUSEADAPTER
MOUSEADAPTER
-dimension : Dimension
-position : String
-dureeActiviteSelected : Float
-heureFinMax : Float
-heureDebutMin : Float
-codeSelected : String
+MouseAdapter(d : Dimension)
+verificationDrop()
+verificationSection()
+verificationPositionHoraire(p : Point)
+jourHeureToActivite(a : Activite)
+getPositionCursor()
1
lISTEACTIVITESDEJAPLACEES
drawing
HORAIREDRAWING
-horaireControler : HORAIRECONTROLER
-initialDimension : Dimension
-radius : int
+HORAIREDRAWING(horaireControler : HORAIRECONTROLER, initialDimension : Dimension)
+draw(g : Graphics) : void
+drawActivite(g : Graphics) : void
+drawHoraire(g : Graphics) : void
+afficherJourHeure() : String

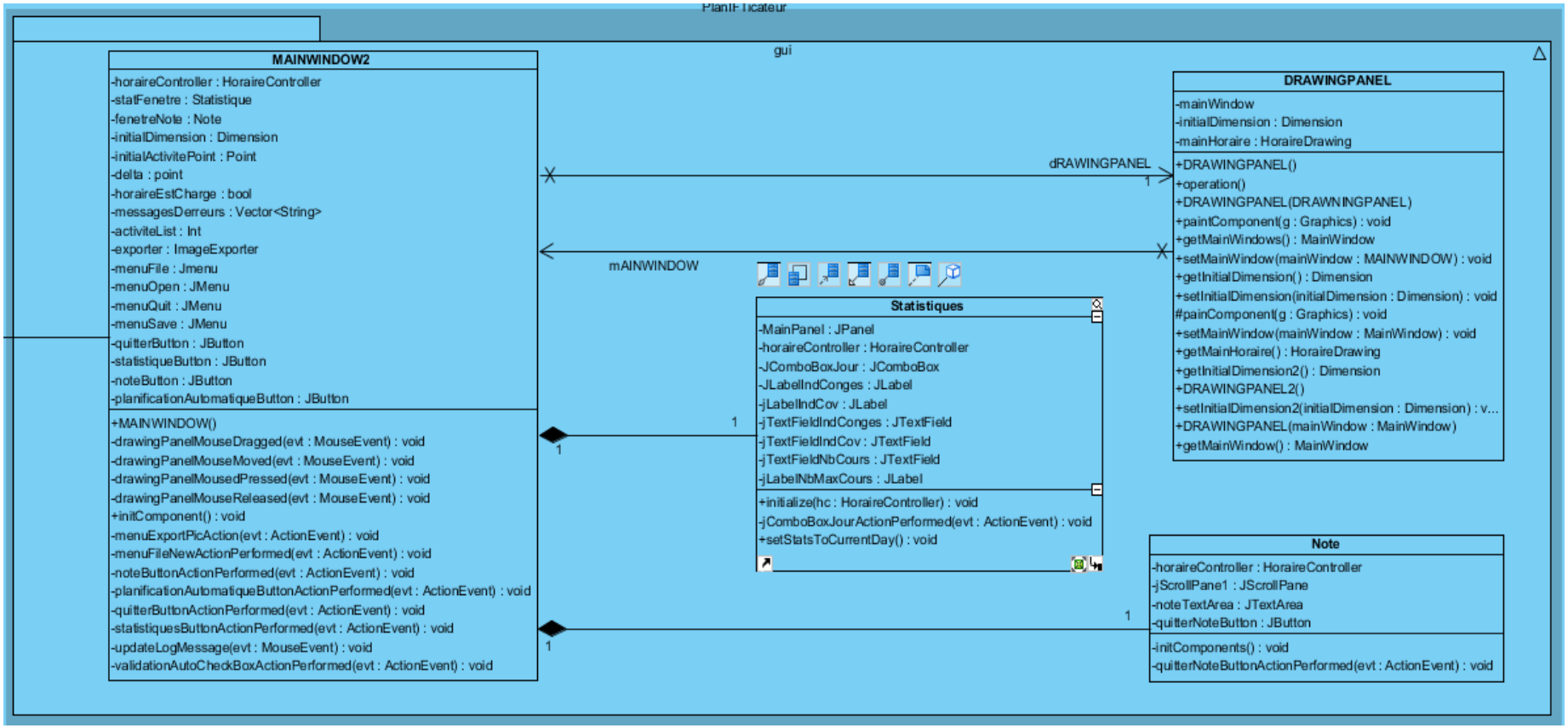
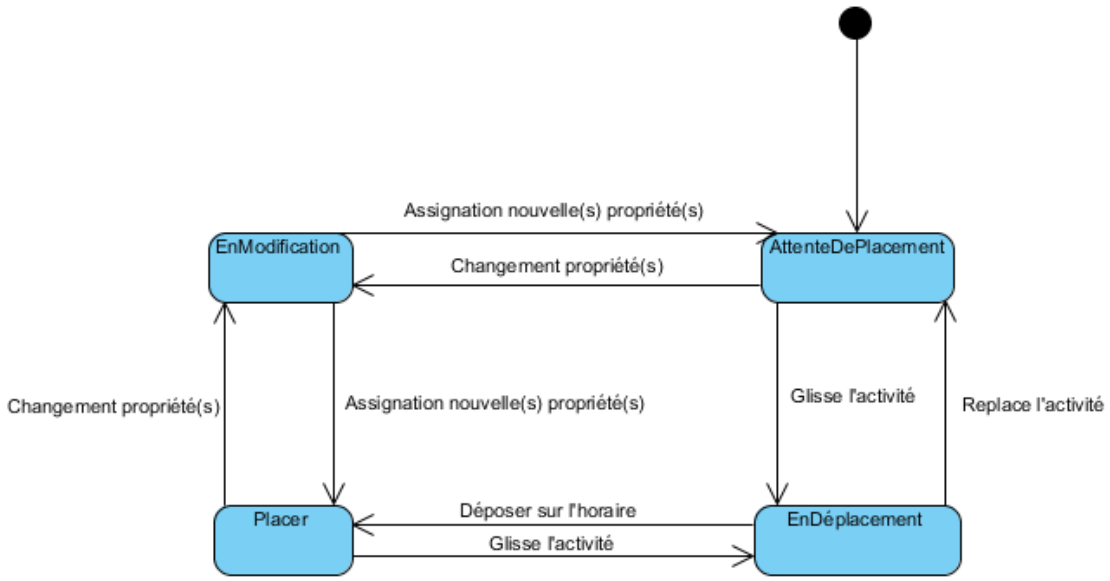

PlanIFicateur
gui
MAINWINDOW2
-horaireController : HoraireController
-statFenetre : Statistique
-fenetreNote : Note
-initialDimension : Dimension
-initialActivitePoint : Point
-delta : point
-horaireEstCharge : bool
-messagesDerreurs : Vector<String>
-activiteList : Int
-exporter : ImageExporter
-menuFile : Jmenu
-menuOpen : JMenu
-menuQuit : JMenu
-menuSave : JMenu
-quitterButton : JButton
-statistiqueButton : JButton
-noteButton : JButton
-planificationAutomatiqueButton : JButton
+MAINWINDOW()
-drawingPanelMouseDragged(evt : MouseEvent) : void
-drawingPanelMouseMoved(evt : MouseEvent) : void
-drawingPanelMousedPressed(evt : MouseEvent) : void
-drawingPanelMouseReleased(evt : MouseEvent) : void
+initComponent() : void
-menuExportPicAction(evt : ActionEvent) : void
-menuFileNewActionPerformed(evt : ActionEvent) : void
-noteButtonActionPerformed(evt : ActionEvent) : void
-planificationAutomatiqueButtonActionPerformed(evt : ActionEvent) : void
-quitterButtonActionPerformed(evt : ActionEvent) : void
-statistiquesButtonActionPerformed(evt : ActionEvent) : void
-updateLogMessage(evt : MouseEvent) : void
-validationAutoCheckBoxActionPerformed(evt : ActionEvent) : void
dRAWINGPANEL
1
mAINWINDOW
1
1
1
1
Statistiques
-MainPanel : JPanel
-horaireController : HoraireController
-JComboBoxJour : JComboBox
-JLabelIndConges : JLabel
-jLabelIndCov : JLabel
-jTextFieldIndConges : JTextField
-jTextFieldIndCov : JTextField
-jTextFieldNbCours : JTextField
-jLabelNbMaxCours : JLabel
+initialize(hc : HoraireController) : void
-jComboBoxJourActionPerformed(evt : ActionEvent) : void
+setStatsToCurrentDay() : void
DRAWINGPANEL
-mainWindow
-initialDimension : Dimension
-mainHoraire : HoraireDrawing
+DRAWINGPANEL()
+operation()
+DRAWINGPANEL(DRAWNINGPANEL)
+paintComponent(g : Graphics) : void
+getMainWindows() : MainWindow
+setMainWindow(mainWindow : MAINWINDOW) : void
+getInitialDimension() : Dimension
+setInitialDimension(initialDimension : Dimension) : void
#painComponent(g : Graphics) : void
+setMainWindow(mainWindow : MainWindow) : void
+getMainHoraire() : HoraireDrawing
+getInitialDimension2() : Dimension
+DRAWINGPANEL2()
+setInitialDimension2(initialDimension : Dimension) : v...
+DRAWINGPANEL(mainWindow : MainWindow)
+getMainWindow() : MainWindow
Note
-horaireController : HoraireController
-jScrollPane1 : JScrollPane
-noteTextArea : JTextArea
-quitterNoteButton : JButton
-initComponents() : void
-quitterNoteButtonActionPerformed(evt : ActionEvent) : void

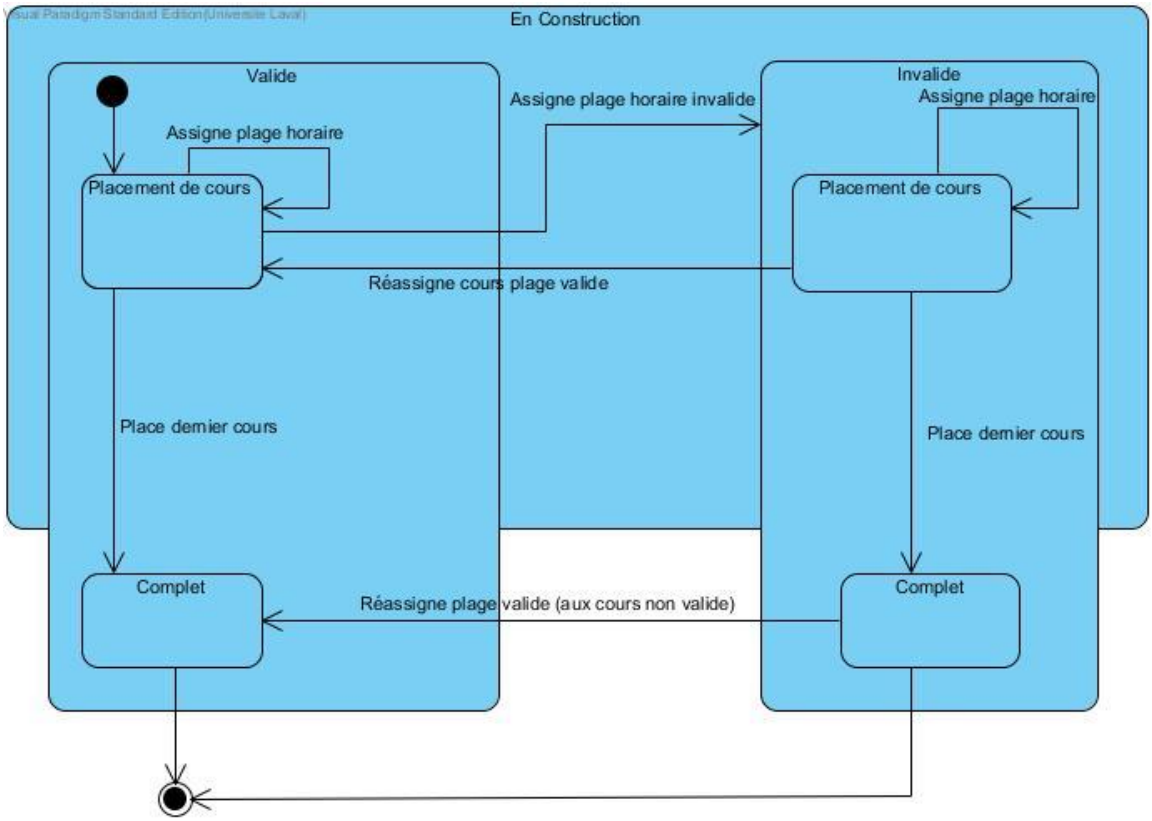
Diagramme d'états

Diagramme d'états d'une activité



Dans ce diagramme, nous pouvons voir les différents états d'un objet **Activité** créé lorsque le programme est en fonction. Le point de départ étant lorsqu'un objet **Activité** est en attente d'être placé. Par la suite, il est possible de modifier certains champs d'une activité et de lui assigner de nouvelles propriétés. Il est possible de faire cela lorsque l'objet **Activité** est placé sur l'horaire ou bien lorsqu'il n'est pas encore placé (soit, en déplacement). Lorsque l'objet **Activité** est sélectionné, il est en mode déplacement (ce mode est indiqué par le changement de couleur de l'objet **Activité**). Le dît objet peut aussi avoir été placé sur l'horaire et remit dans la liste d'activités en attente d'être assigné à une plage horaire, encore une fois, en passant par l'état *AttenteDePlacement*.

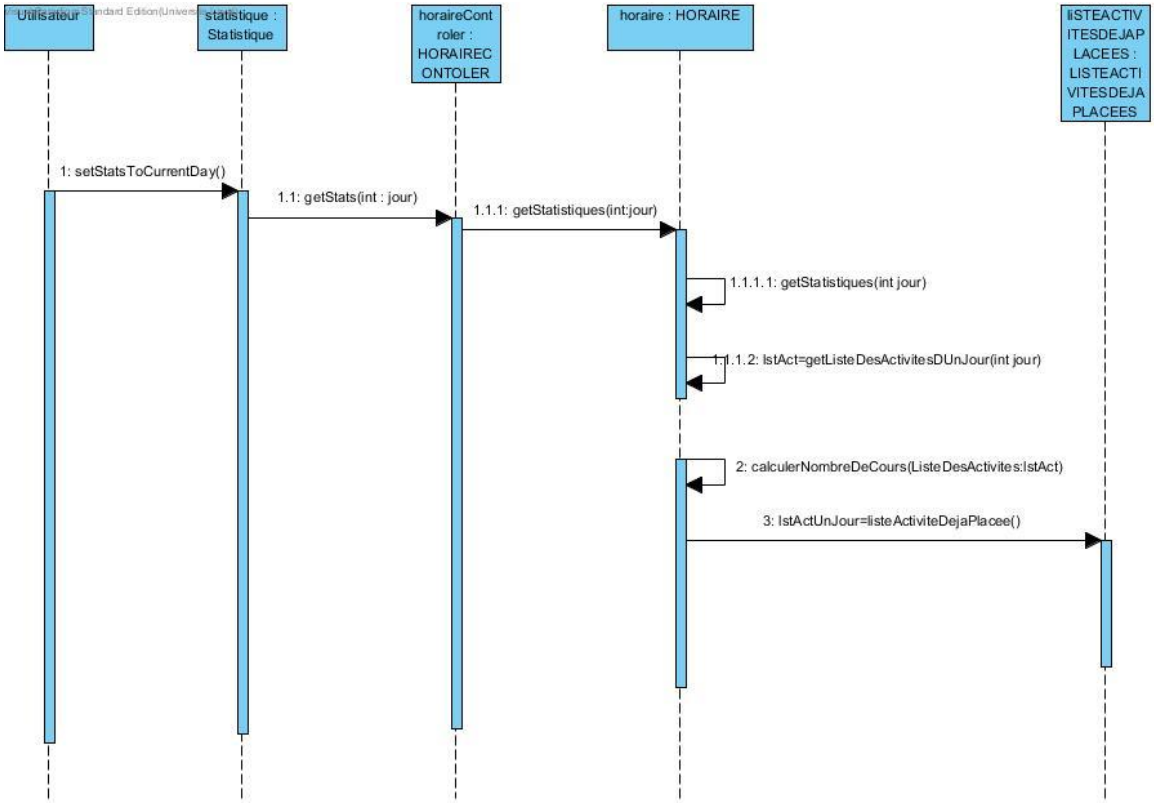
Diagramme d'états d'un horaire



Dans le diagramme ci-dessus, nous constatons que l'horaire peut adopter plusieurs états simultanément. Tout d'abord, lors que l'on fait l'ouverture d'un horaire, il est en construction. Un horaire peut être soit valide ou non. Dans notre cas, un horaire est valide lorsqu'il n'y a pas de conflit d'horaire pour les activités qui soit, appartiennent à la même grille de cheminement, soit un conflit par rapport aux restrictions de cette activité. Nous avons fait ce choix, puisque l'indicateur de validité prend en considération seulement les conflits pour afficher un horaire qui est valide. Cependant, un horaire que l'on doit considérer valide est un horaire qui est dit complet, sans conflit d'horaire et dont toutes les activités sont placées.

Diagrammes de séquence

Indicateur du nombre de cours par jour



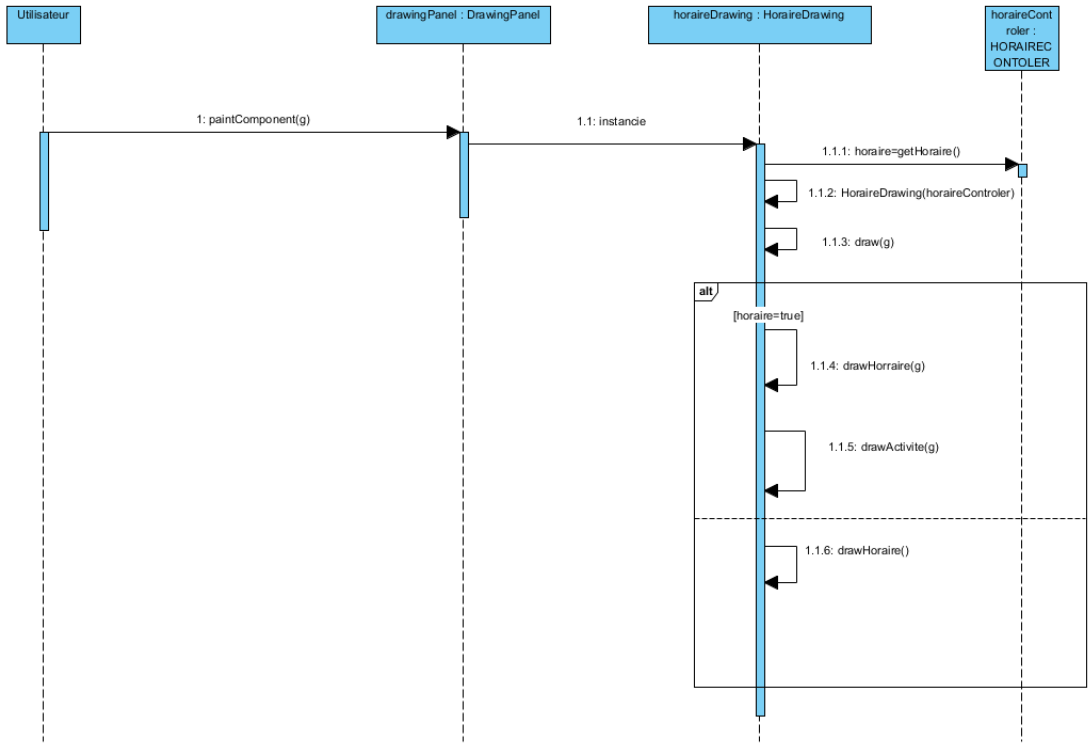
Lorsque l'utilisateur clique sur le bouton **Statistiques**, une fenêtre est instanciée avec les dimensions et les champs adéquats. La méthode **setStatsToCurrentDay()** de **Statistiques** est invoquée. **Statistiques** invoque la methode **getStats(jour)** du contrôleur en lui passant un jour en paramètre. Le contrôleur appelle ensuite **getStatistique(jour)** de l'horaire. L'horaire appelle respectivement ses méthodes :

-getStatistiques(jour);

-lstAct=getListeDesActivitesDUnJour(jour);

-calculerNombreDeCours(ListeDesActivites).

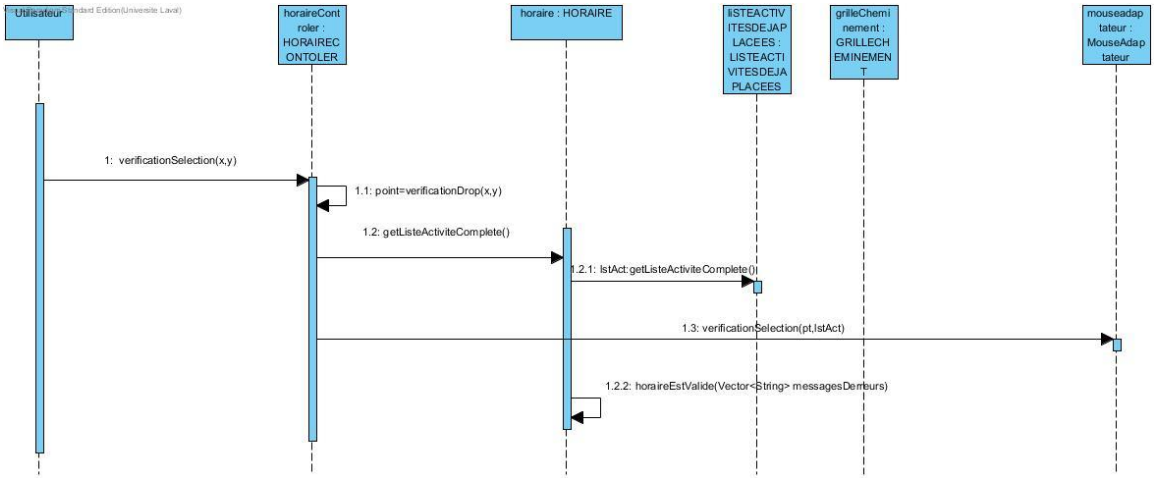
Affichage de la grille horaire



L'utilisateur appelle la méthode **paintComponent(g)** de **DrawingPanel** en passant un objet graphique. **DrawingPanel** instancie un objet **HoraireDrawing** qui invoque à son tour **getHoraire** du contrôleur qui retourne un booléen. Selon la valeur de ce booléen, la méthode **draw(g)** invoque à la fois **drawHoraire(g)** et, soit **drawActivite(g)**, soit **drawActivite(g)**.

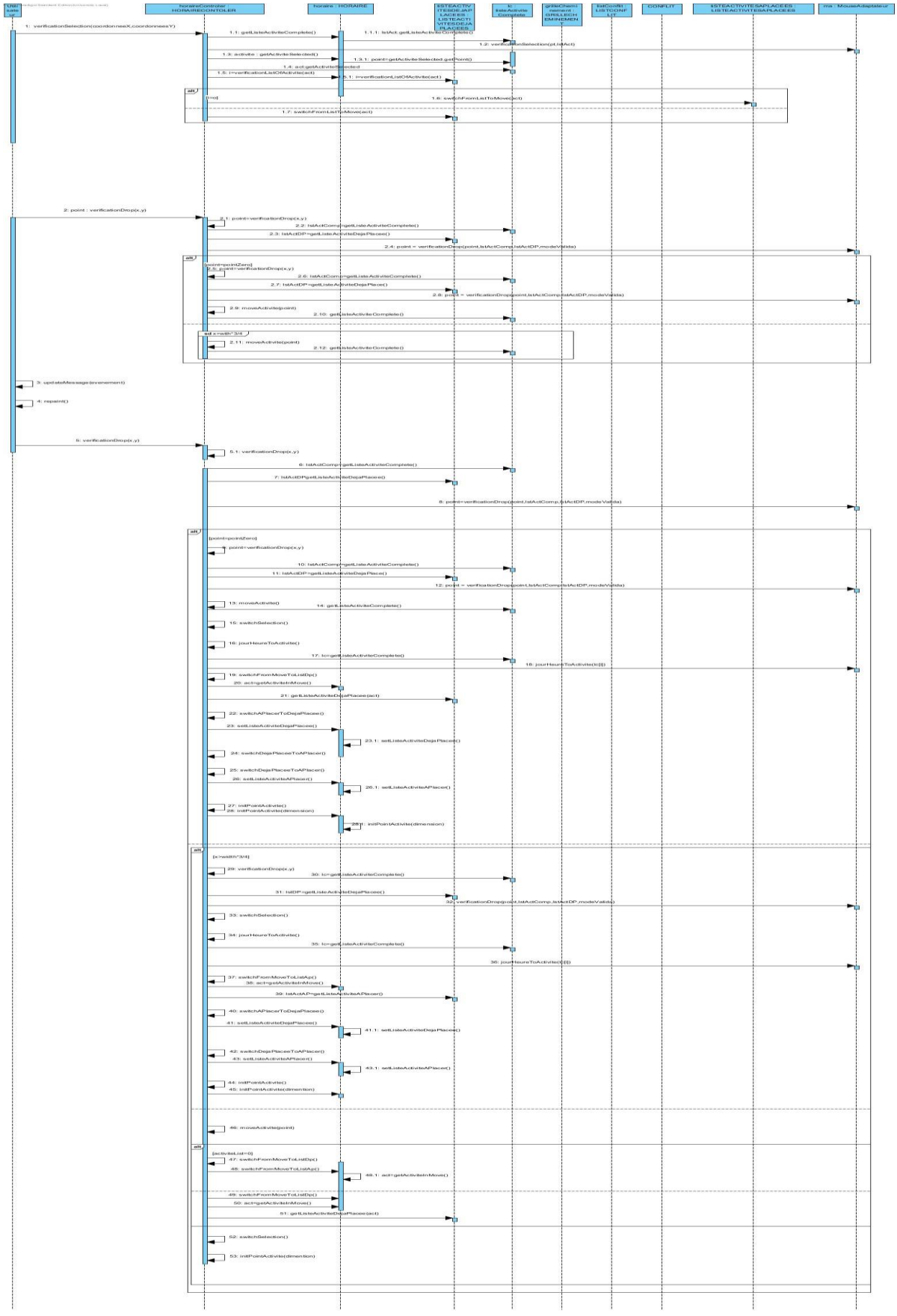
Validation qu'un jour/heure de début est valide pour un cours donné

Le diagramme a été modifié pour correspondre aux classes et fonctions de notre application.



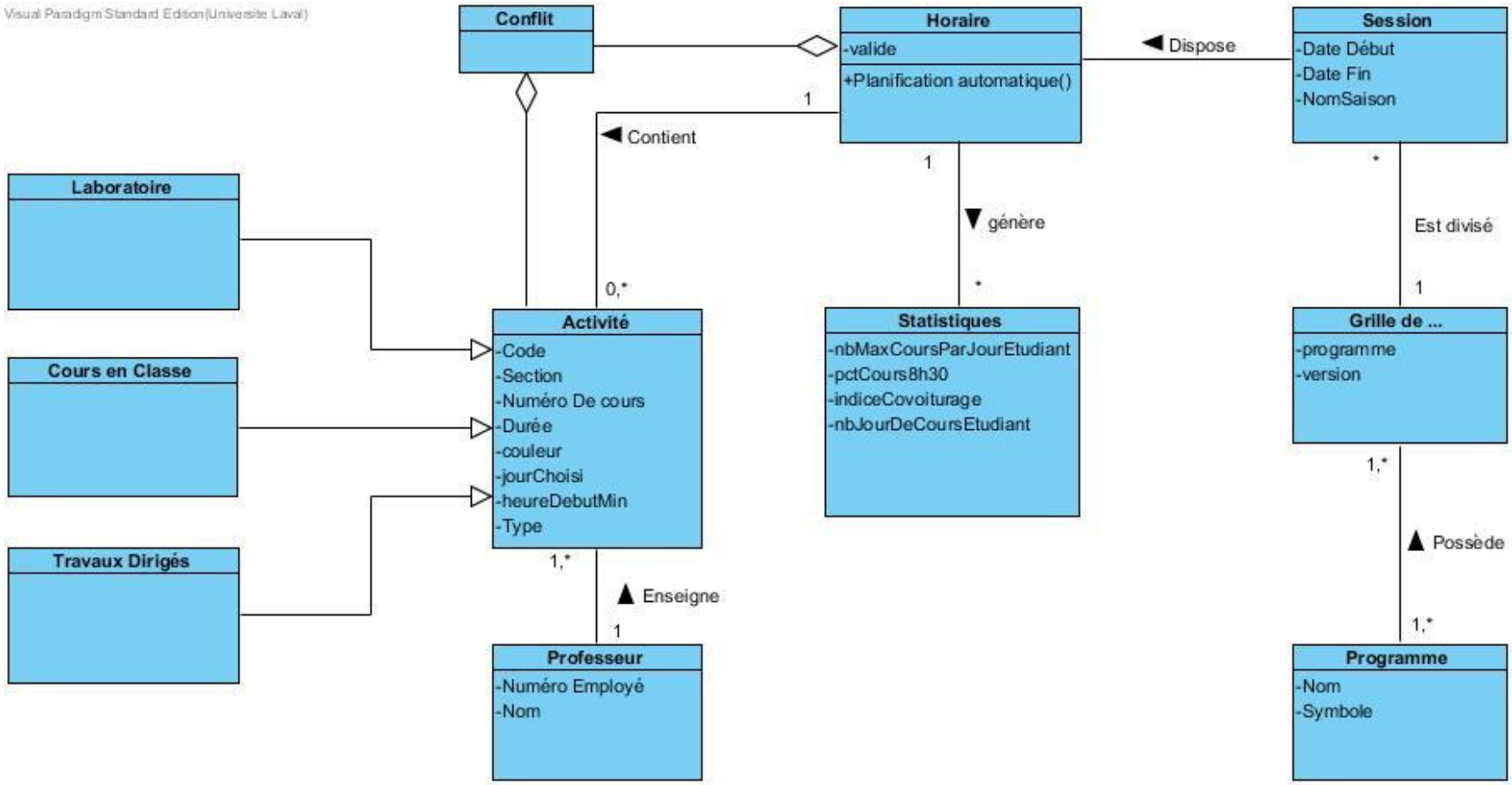
Sélection d’un cours dans la grille horaire avec l’aide de la souris / déplacement d’un cours dans la grille horaire

Étant donné la taille du diagramme, nous avons inclus une image exportée de *Visual Paradigm* en pièce jointe avec le rapport. Vous pouvez aussi l’ouvrir directement dans *Visual Paradigm*.

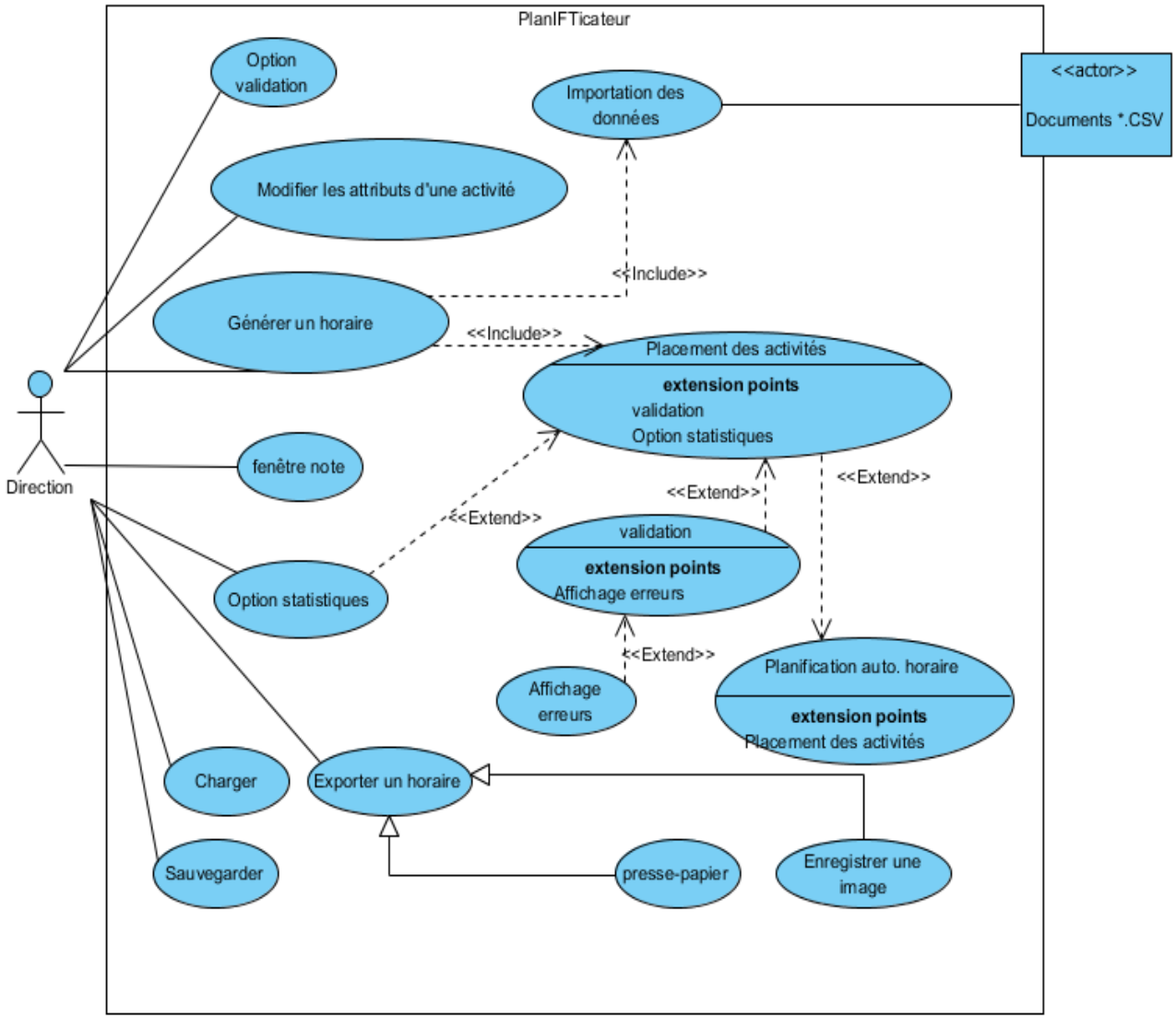

horaireControler : HORAIRECONTOLER
horaire : HORAIRE
listeActivite Complete
1: verificationSelection(coordonneeX,coordonneeY)
1.1: getListeActiviteComplete()
1.2: verificationSelection(pt,lstAct)
1.3: activite : getActiviteSelected()
1.4: act:getActiviteSelected
1.5: i=verificationListOfActivite(act)
alt
[no]
1.6: switchFromListToMove(act)
1.7: switchFromListToMove(act)
2: point : verificationDrop(x,y)
2.1: point=verificationDrop(x,y)
2.2: lstActComp=getListeActiviteComplete()
2.3: lstActDP=getListeActiviteDejaPlacee()
2.4: point = verificationDrop(point,lstActComp,lstActDP,modeValida)
[point=pointZero]
2.5: point=verificationDrop(x,y)
2.6: lstActComp=getListeActiviteComplete()
2.7: lstActDP=getListeActiviteDejaPlacee()
2.8: point = verificationDrop(point,lstActComp,lstActDP,modeValida)
2.9: moveActivite(point)
2.10: getListeActiviteComplete()
sd x=with*3/4
2.11: moveActivite(point)
2.12: getListeActiviteComplete()
3: updateMessage(evenement)
4: repaint()
5: verificationDrop(x,y)
5.1: verificationDrop(x,y)
6: lstActComp=getListeActiviteComplete()
7: lstActDP=getListeActiviteDejaPlacee()
8: point=verificationDrop(point,lstActComp,lstActDP,modeValida)
[point=pointZero]
9: point=verificationDrop(x,y)
10: lstActComp=getListeActiviteComplete()
11: lstActDP=getListeActiviteDejaPlacee()
12: point = verificationDrop(point,lstActComp,lstActDP,modeValida)
13: moveActivite()
14: getListeActiviteComplete()
15: switchSelection()
16: jourHeureToActivite()
17: lc=getListeActiviteComplete()
18: jourHeureToActivite(lc[i])
19: switchFromMoveToListDp()
20: act=getActiviteInMove()
21: getListeActiviteDejaPlacee(act)
22: switchAPlacerToDejaPlacee()
23: setListeActiviteDejaPlacee()
23.1: setListeActiviteDejaPlacee()
24: switchDejaPlaceeToAPlacer()
25: switchDejaPlaceeToAPlacer()
26: setListeActiviteAPlacer()
26.1: setListeActiviteAPlacer()
27: initPointActivite()
28: initPointActivite(dimension)
28.1: initPointActivite(dimension)
alt
[x>with*3/4]
29: verificationDrop(x,y)
30: lc=getListeActiviteComplete()
31: lstDP=getListeActiviteDejaPlacee()
32: verificationDrop(point,lstActComp,lstActDP,modeValida)
33: switchSelection()
34: jourHeureToActivite()
35: lc=getListeActiviteComplete()
36: jourHeureToActivite(lc[i])
37: switchFromMoveToListAp()
38: act=getActiviteInMove()
39: lstActAP=getListeActiviteAPlacer()
40: switchAPlacerToDejaPlacee()
41: setListeActiviteDejaPlacee()
41.1: setListeActiviteDejaPlacee()
42: switchDejaPlaceeToAPlacer()
43: setListeActiviteAPlacer()
43.1: setListeActiviteAPlacer()
44: initPointActivite()
45: initPointActivite(dimention)
46: moveActivite(point)
alt
[activiteList=0]
47: switchFromMoveToListDp()
48: switchFromMoveToListAp()
48.1: act=getActiviteInMove()
49: switchFromMoveToListDp()
50: act=getActiviteInMove()
51: getListeActiviteDejaPlacee(act)
52: switchSelection()
53: initPointActivite(dimention)

Annexe

Modèle du domaine



Modèle des cas d'utilisation



Glossaire

Algorithme

Suite de règles permettant de résoudre un problème.

Application

Programme ou ensemble de programmes visant à aider un utilisateur d'un ordinateur dans le traitement d'une tâche précise. *(Réf. : Larousse.fr)*

Cas d'utilisation

Relation établie entre une fonctionnalité et, soit un acteur ou une autre fonctionnalité.

Chemin

Emplacement précis sur un disque dur définit par la suite ordonnée des dossiers pour avoir accès au fichier recherché.

Diagramme de séquence système (DSS)

Représentation des interactions chronologiques entre un système et ses acteurs. *(réf. : Wikipédia)*

Domaine d'affaires

Réfère pour l'ensemble des processus d'affaires d'un projet ou d'une organisation. Englobe les entités, les acteurs et autres participants de ces processus.

« Drag and drop »

(Terme anglais) En informatique, processus durant lequel un usager d'un ordinateur sélectionne un objet, au moyen d'une souris, en maintenant enfoncé le bouton gauche de cette dernière afin de déplacer le dît objet et de le relâcher à l'endroit voulu en relâchant également le bouton de sa souris.

Fenêtre

Zone d'affichage d'information d'un programme.

Fonction

Bloc d'une séquence d'instructions visant un but, une fonctionnalité précise.

Logiciel

Ensemble d'instructions et de règles interprétables par un ordinateur.

Multiplateforme

Fonctionnant sur plusieurs plateformes, soit plusieurs ordinateurs/systèmes d'exploitation différents.

Multiutilisateur

Offrant la possibilité à plusieurs usagers d'interagir simultanément sur le même logiciel.

PlanIFTicateur

Nom du logiciel en développement pour le projet contenu dans ce document.

Plateforme PC

Liaison entre un ordinateur personnel munit d'un processeur spécifique et du système d'exploitation *Windows*.

Presse-papier

Fonction intégrée dans tous les systèmes d'exploitation stockant des données que l'on souhaite déplacer ou copier.

Programme

Succession d'instructions qu'un ordinateur peut exécuter afin d'accomplir des opérations.

Serveur dédié

Système informatique dont l'ensemble des ressources est dédié à un seul utilisateur.

Serveur web

Système informatique qui a pour fonction la publication de sites web à la demande d'un autre système.

Session

Période de 15 semaines durant laquelle un étudiant de l'université doit accomplir les objectifs de ses cours.

GUI

Signifie : **G**raphical **U**ser **I**nterface, ce qui représente l'interface graphique utilisateur. C'est ce qui est affiché par le programme.

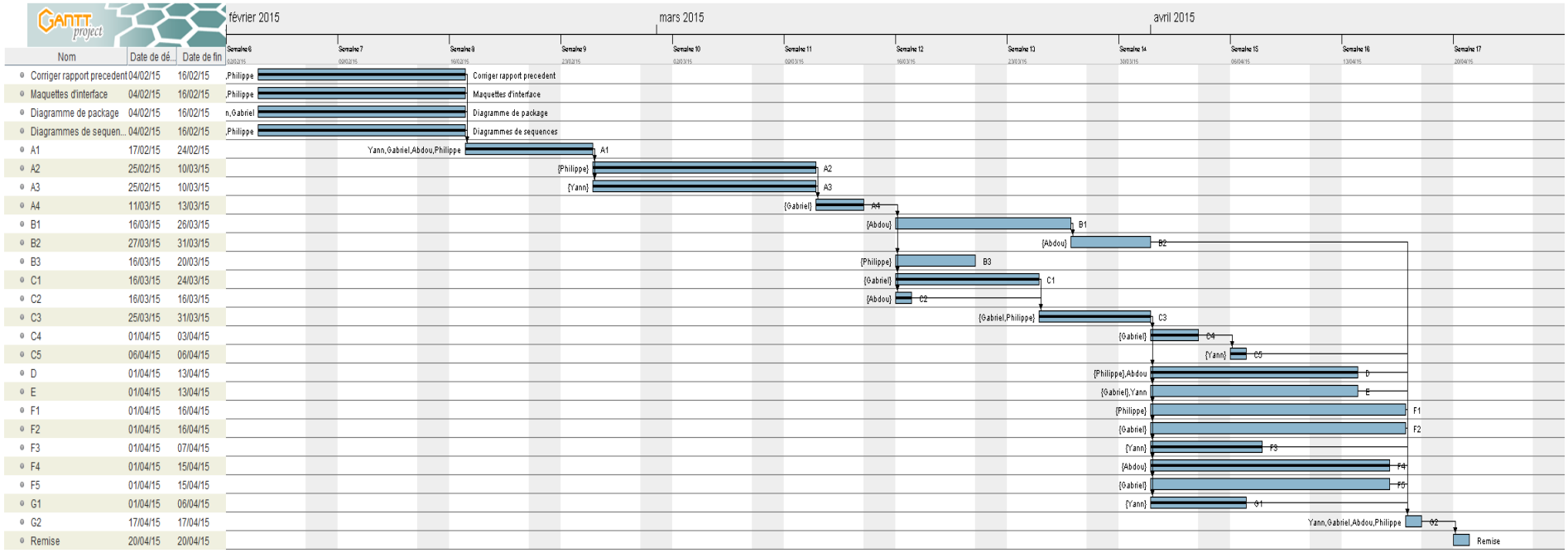
Diagramme de package

Permet de représenter l'architecture des différents groupements de classe en différentes couches logiques.

Diagramme de séquence

Diagramme illustrant ce qui se produit lors de certaines actions. Permet de comprendre la communication entre les classes.

Gestion de projet



Voir diagrammeDeGant.gan