TP3 - Récupération d'architecture et architectures modulaires

Groupe Mike

Achille Bineli (1664291)

Fabrice Dugas (1690694)

Thomas Thebaud (1974069)

Quoc-Hao Tran (1972967)

LOG8430

9 novembre 2018

Question 1 : Récupérez l'architecture de MiniDraw. (60pts)

Suite à un bug avec le projet odem2mdg, certaines classes sont absentes lors de l'extraction de l'architecture avec Bunch. Nous avons donc pris certaines décisions stratégiques afin de trouver le meilleur paquet pour chacune de ces classes orphelines.

- **GroupFigure** → **minidraw.framework.Figure.ss** puisque GroupFigure implémente l'interface Figure.
- **BoardActionTool** → **minidraw.boardgame.BoardDrawing.ss** pour améliorer la cohésion puisque cette classe utilise d'autres classes de ce paquet.

Nous avions initialement inclut le code source des paquets **demo.***, mais suite à une discussion avec le chargé de laboratoire, nous avons compris qu'il ne fallait faire l'exercice qu'avec le code source de minidraw et non pas le code utilisé pour faire une démonstration de minidraw. Pour cette raison, certaines classes supplémentaires apparaissent dans l'architecture extraite avec Bunch telles que Breakthrough, BreakthroughPieceFactory, ChessboardBackgroundFactory, etc. Afin de respecter l'exercice, nous avons retiré ces classes des paquets lors de la refactorisation par module pour la question 2.

Question 2 : Migrez MiniDraw vers une architecture modulaire. (40pts)

FactoryFacade pour le paquet **minidraw.framework.Factory.ss**:
Cette façade contient principalement des méthodes afin d'initialiser MiniDrawApplication et d'utiliser l'interface Factory. Les classes qui utilisent cette façade sont Breakthrough, LogoPuzzle, Mutliview, ShowCompositeFigure, ShowFigures et ShowRectangle. En général, le but est atteint, soit seule la classe façade est importée à part dans le cas des classes qui implémentent l'interface Factory directement telle que BreakthroughFactory, PuzzleFactory et EmptyCanvasFactory.

AbstractToolFacade pour **minidraw.standard.AbstractTool.ss:**
Un des couplages afférents est causé par l'utilisation de **RubberBandSelectionStrategy** dans la classe **SelectionTool**. Ce couplage peut être enlevé grâce au patron façade en gérant toute interaction avec l'interface. Par contre, les autres couplages afférents sont causés par des classes extensions de la classe **AbstractTool** telles que **DragTracker** et **SelectionTool** qui ne peuvent être réglés facilement grâce au patron façade.

FigureFacade pour **minidraw.standard.Figure.ss:**
Un des couplages afférents est causé par l'utilisation de **StandardDrawing** et de **GroupFigure** dans la classe **LogoPuzzle** et **ShowCompositeFigure**. Ce couplage peut être enlevé grâce au patron façade en gérant toute interaction avec l'interface. Par contre, les autres couplages afférents sont causés par des classes extensions de la classe **StandardDrawing** telles que **BoardDrawing** qui ne peuvent être réglés facilement grâce au patron façade.

BoardDrawingFacade pour **minidraw.boardgame.BoardDrawing.ss**
Dans le cas du paquet **minidraw.boardgame.BoardDrawing.ss**, le couplage afférent est causé par l'instanciation des classes BoardActionTool, BoardFigure et BoardGameObserver<Position>.

StandardDrawingChangeListenerHandlerFacade pour **minidraw.standard.handlers.StandardDrawingChangeListenerHandler.ss**:
Cette façade gère les appels aux classes StandardDrawingChangeListenerHandler et DrawingChangeEvent. Elle sert donc principalement de gestionnaire d'événements (ajout et retrait de listener et envoie d'événements).

DrawingEditorFacade pour **minidraw.framework.DrawingEditor.ss:**
Cette façade encore une fois gère principalement la création des objets de ce paquet. Il serait donc probablement plus pertinent de nommer cette classe *DrawingEditorFactory* puisqu'il ressemble beaucoup plus au patron Usine que Façade.