AspectOCL Constraints- Case Study 2

AspectOCL Constraint 1:

```
mapping mapSudokuUnique
{
        let T -> A : {    RegisteredUser  -> RegisteredUser :: username
                          Sudoku  ->  Sudoku :: id_sudoku
                      }
}
aspect UniquenessSudoku
{
        import_mapping mapSudokuUnique
        pointcut uniqueSudoku
        context T
        intro:
        inv uniqueSudoku
        T -> allInstances() -> isUnique(A)

}
```

AspectOCL Constraint 2:

```
mapping mapSudokuSize
{
        let T -> A : {    Sudoku  -> Row :: row_num
                          Sudoku  ->  Column :: col_num
                          RowCell  ->  Cell:: correct_value
                          ColumnCell -> Cell:: correct_value
                          Region  -> Cell:: correct_value
                    }
}
```

```
aspect SizeOfSudoku
{
        import_mapping mapSudokuSize
        pointcut SizeSudoku
        context T
        intro:
        inv SizeSudoku
        T.allInstance() -> select(s| s. A  <> self. A) -> size()=1

}
```

AspectOCL Constraint 3:

```
mapping mapIndividualEffect
{
        let T -> {S,A} : { NewPlayer :: effect()  ->
                                        Player :: play_att , Player }
                           NewAdministrator :: effect()  ->
                                        Administrator :: admin , Administrator }
                         }
}

aspect EffectOfIndividual
{
        import_mapping mapIndividualEffect
        pointcut effect
        context T
        intro:
        post effect
        S.oclIsNew() and oclIsTypeOf(A) and UserHasAttributes(S)

}
```

AspectOCL Constraint 4:

```
mapping mapfinishedStatus
{
        let T : {       SudokuChoice:: UnfinishedSudoku() : Boolean,
                        GameMove:: UnfinishedSudoku() : Boolean,
                        IncorrectCellsCheck:: UnfinishedSudoku() : Boolean,
                        CompoundGameMove:: UnfinishedSudoku() : Boolean
                }
}
aspect finishedSudoku
{
        import_mapping mapfinishedStatus
        pointcut selectsudokoFinish
        context T
        intro:
        pre selectsudokoFinish
        sudoku.finished = false
}
```

AspectOCL Constraint 5:

```
mapping mapMoveEffect
{
        let T -> {A, B, S}: {   Undo:: effect() ->
                                        {Undo, UndoMove, Sudoku  ::  lastDisposableMove }
                                Redo:: effect() ->
                                        {Redo,  RedoMove, Sudoku  ::  lastUndoneMove }
                            }
}
aspect SudokuEffect
{
        import_mapping mapMoveEffect
        pointcut selectEffectTarget
```

context T

intro:

post selectEffectTarget

A. oclIsNew() and A.oslIsTypeOf(B) and A. nonPredefinedCell = sudoku. S

Undo. oclIsNew() and undo.

}

AspectOCL Constraint 6:

mapping mapMailStatus

{

let T : { NewRegisteredUser:: CorrectMail() : Boolean

MailUpdate:: CorrectMail() : Boolean

}

}

aspect MailCorrection

{

import_mapping mapMailStatus

pointcut Mail

context T

intro:

body Mail

mail. CorrectMail()

}

AspectOCL Constraint 7:

mapping mapCurrentPlayer

{

let T : { SudokuChoice:: SudokuIsTheCurrentOfPlayer() : Boolean

IncorrectCellsCheck:: SudokuIsTheCurrentOfPlayer() : Boolean

CompoundGameMove:: SudokuIsTheCurrentOfPlayer() : Boolean

}

}

```
aspect CurrentGamePlayer
{
        import_mapping mapCurrentPlayer
        pointcut GamePlayer
        context T
        intro:
        body GamePlayer
        player.currentSudoku = Sudoku and sudoku.finished = false

}
```

AspectOCL Constraint 8:

```
mapping mapCellValue
{
        let T :   {       PutValueInACell:: CellsPartOfCurrentSudoku() : Boolean
                          CellCheck:: CellsFromCurrentSudoku() : Boolean
                  }
}

aspect PredefinedCellValue
{
        import_mapping mapCellValue
        pointcut CellValue
        context T
        intro:
        body CellValue
        player.currentSudoku.nonPredefinedCell -> includes (nonPredefinedCell)

}
```