Phần 1.

1) Game requirement Analyst

2) Use-case Diagram

3) Conceptual class and their association

4) Incomplete class diagram

5) Focus-usecase

6) Activity diagram

7) Sequence diagram

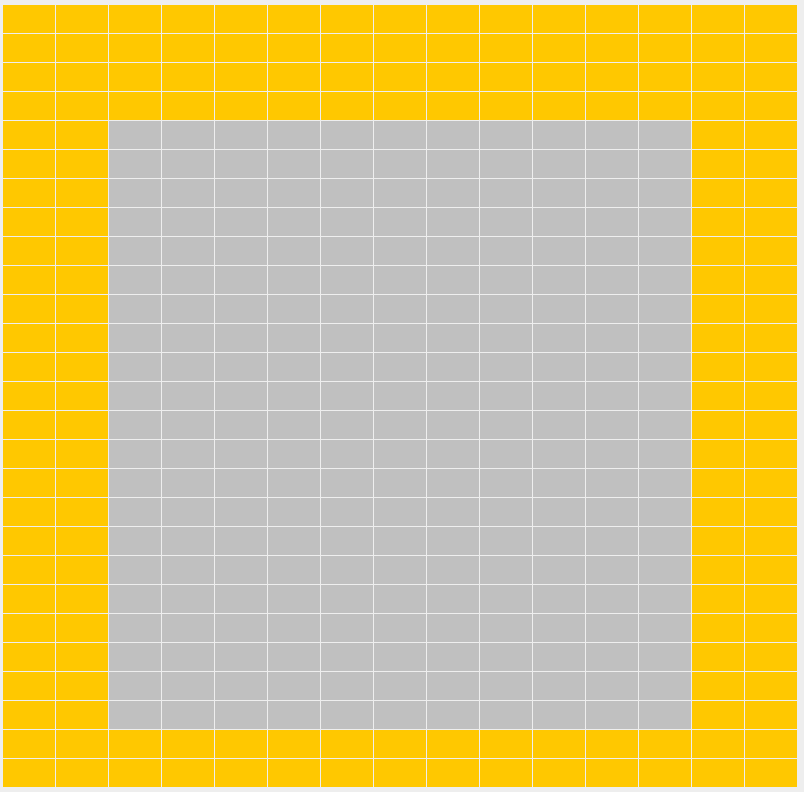
8)Complete class diagram

Phần 2.  
1) Implement

Phần 1 )   
1) Game requirement Analyse  
Tetris là Trò chơi có bảy loại **khối hình**: I (thẳng đứng), J, L, O (vuông), S, T, Z. Ta thấy mỗi khối gạch được cấu tạo từ 4 **hình vuông nhỏ** xếp lại với nhau. Ta có thể coi các khối gạch đó như là những hình chữ nhật có kích thước khác nhau.

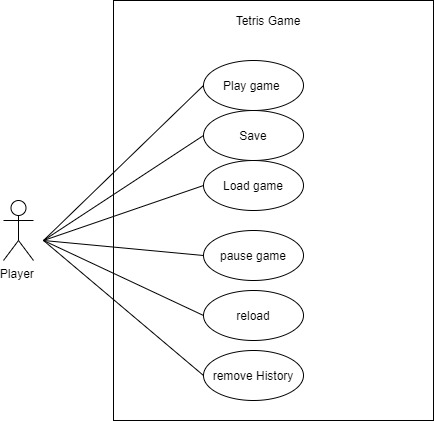
Các hình khác được tạo ra khi xoay các khối cơ bản này các góc tương ứng 90 độ, 180 độ, 270 độ.

Một chuỗi ngẫu nhiên của Tetriminos rơi xuống sân chơi (một trục đứng hình chữ nhật, được gọi là "tốt" hay "ma trận").  
  
Mục tiêu của trò chơi là di chuyển các khối gạch đang rơi từ từ xuống trong kích thước **hình chữ nhật** 20 hàng x 11 cột (trên màn hình) sao cho khối gạch không rơi xuống được nữa thì trò chơi kết thúc



Chú thích: in đậm là các đối tượng cần thiết kế.  
  
2) Use-case Diagram

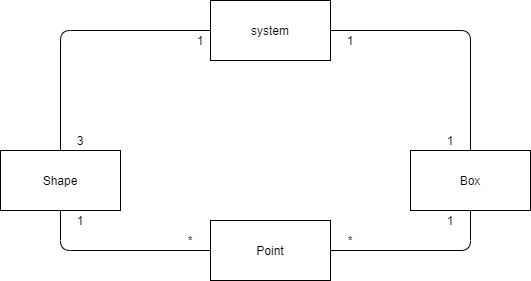
-Từ các ý đã được phân tích trong yêu cầu trò chơi ta có thể hình dung ra nhiều chức năng có cả lớn và nhỏ xuất hiện trong trò chơi tetris này



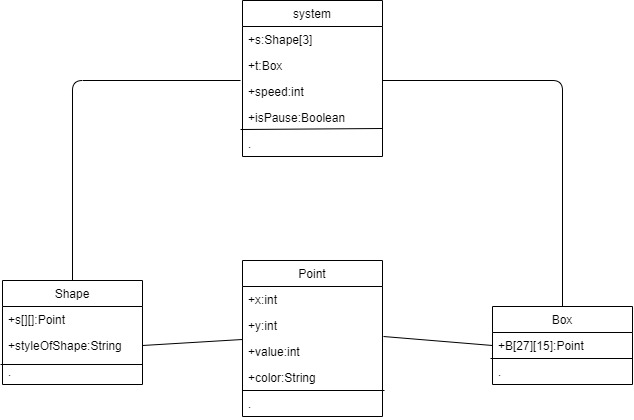
Giải thích: Play game( chức năng quan trọng nhất): đây là chức năng để người dùng chơi 1 trận tetris mới.  
 Save game: đúng như tên gọi của nó đây là chức năng để người dùng lưu lại bố cục hiện tại trận đấu.  
 Load Game: đây là 1 chức năng để người dùng chơi tiếp tục 1 trận đấu đã được lưu trước đó  
 Remove History: đây là 1 chức năng để người dùng xóa trận đấu đã lưu trước đó   
 Pause Game( chức năng nhỏ): đây là chức năng để người dùng dừng trận đấu lại  
 Reload Game( chức năng nhỏ):đây là chức năng để người dùng hủy bỏ trận đấu hiện tại và chơi 1 trận đấu mới.

3. Conceptual class and their association

- Từ những dữ liệu đã phân tích trong yêu cầu trò chơi ta có sơ đồ lớp mức quan niệm và mối quan hệ của chúng như sau:



4. Incomplete class diagram  
- ta đã biết tetris là game 2D điều đó đồng nghĩa với việc mỗi ô vuông sẽ là 1 tọa độ trong OXY từ bổ xung điều đó vào sơ đồ lớp mức quan niệm ta có được sơ đồ lớp không hoàn chỉnh như sau:



5)Focus Use-case  
-Đến đây ta có thể mô tả những gì mà mỗi 1 chức năng sẽ thực hiện .

1. Play game (new)

|  |  |
| --- | --- |
| Actor | System |
| 1)Actor request to play game |  |
|  | 2)System load game by the way create new box, list shape( 3 shape random) |
|  | 3) system set location for shape1 in box |
|  | 4) system let’s shape1 drop down and set location of shape1 in box again. Then system wait a while(==speed of game). |
| 5)Actor has ability to control shape move to direct she/he want. |  |
|  | 6)The system check collision before move shape if there is collision, do nothing. Otherwise move to direct actor want. |
|  | 7) system override all full lines by all line Above those those when the shape1 in deepest. |
|  | 8) if system still wait return step 4. Otherwise go to step 10. |
|  | 9)system check full a cell(check lose) if full a cell save record into database(rule1) usecase end here. Otherwise update shape1 = shape2 , shape2= shape3 and create shape3 as a new shape( random). Return step 3. |

2)Load game (continue)

|  |  |
| --- | --- |
| Actor | System |
| 1)Actor request to load game |  |
|  | 2)System load game by the way load box, list shape in database |
|  | 3) system set location in database for shape1 in box |
|  | 4) system let’s shape1 drop down and set location of shape1 in box again. Then system wait a while(==speed of game). |
| 5)Actor has ability to control shape move to direct she/he want. |  |
|  | 6)The system check collision before move shape if there is collision, do nothing. Otherwise move to direct actor want. |
|  | 7) system override all full lines by all line Above those those when the shape1 in deepest. |
|  | 8) if system still wait return step 4. Otherwise go to step 10. |
|  | 9)system check full a cell(check lose) if full a cell save record into database(rule1) usecase end here. Otherwise update shape1 = shape2 , shape2= shape3 and create shape3 as a new shape( random). Return step 3. |

3)Remove history

|  |  |
| --- | --- |
| Actor | System |
| 1)Actor see hitory |  |
|  | 2)system show all 5-best record |
| 3)Actor request to remove history |  |
|  | 4)system remove all database. Usecase end here |

4)Save game

|  |  |
| --- | --- |
| Actor | System |
| 1)Actor are playing game |  |
| 2)Actor request to save game |  |
|  | 3)system save data of all shape in list,data of box into database. Usecase end here |

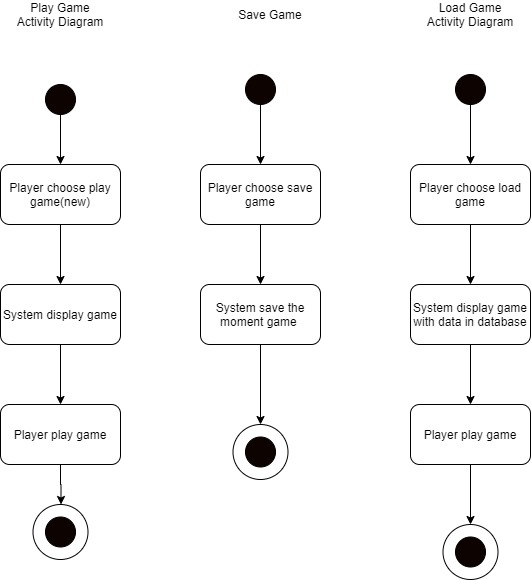
5)pause game

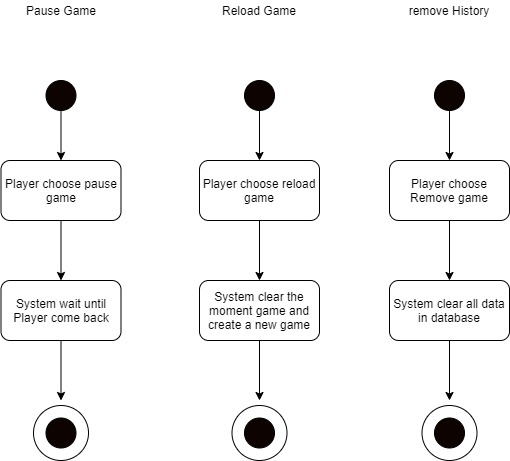
|  |  |
| --- | --- |
| Actor | System |
| 1)Actor are playing game |  |
| 2)Actor request to pause game |  |
|  | 3)system sleep until actor request to resume game |

6)reload game

|  |  |
| --- | --- |
| Actor | System |
| 1)Actor are playing game |  |
| 2)Actor request to reload game |  |
|  | 3)system create a new list shape and new box  Use-case end here. |

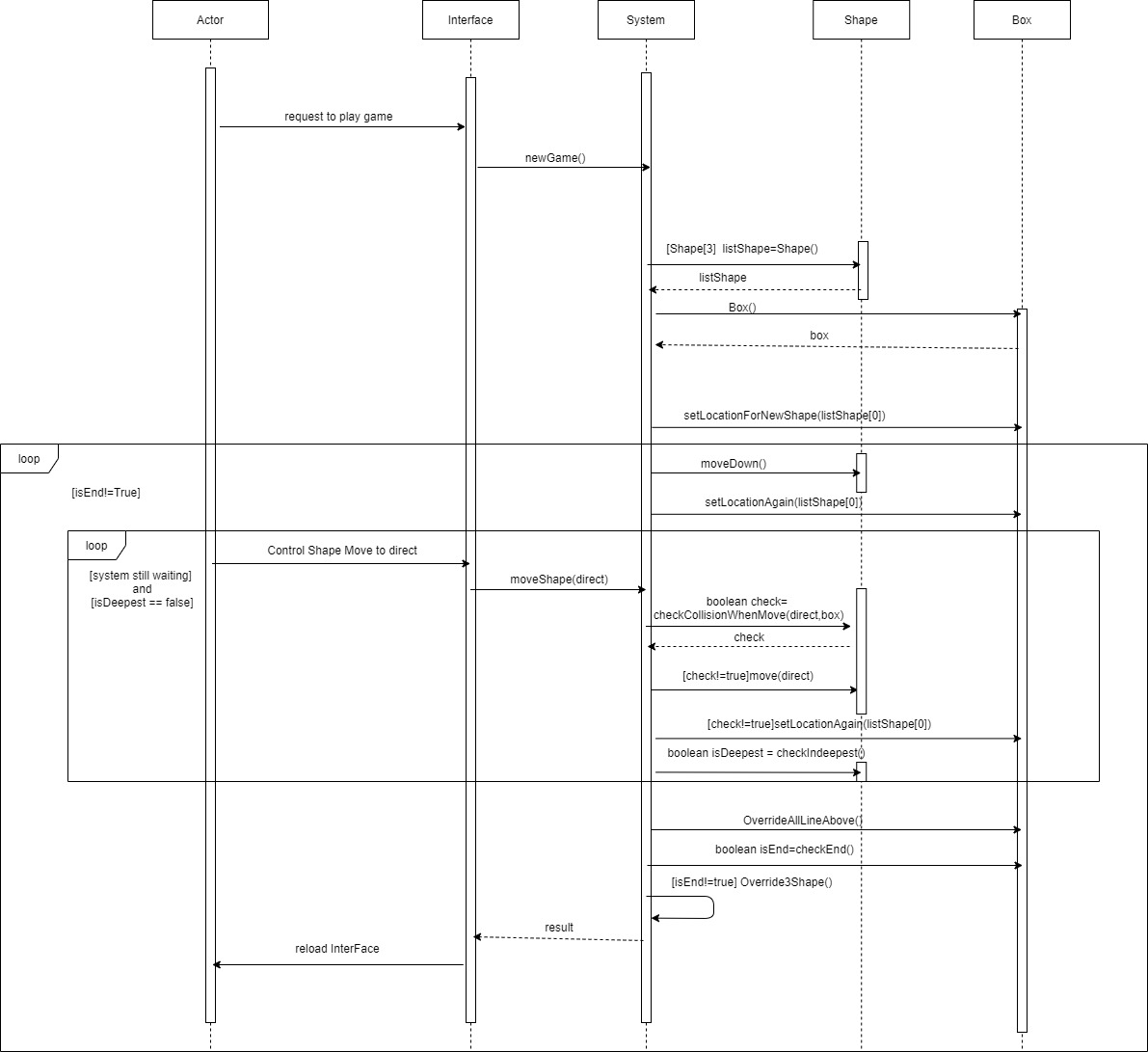
6) Activity diagram  
- Dựa vào focus usecase thì ta đã biết những chức năng sẽ làm gì và làm như thế nào, lúc này ta có thể vẽ activity diagram cho mỗi chức năng nhằm tổng quát hóa để biết được mỗi chức năng sẽ hoạt động ra sao.



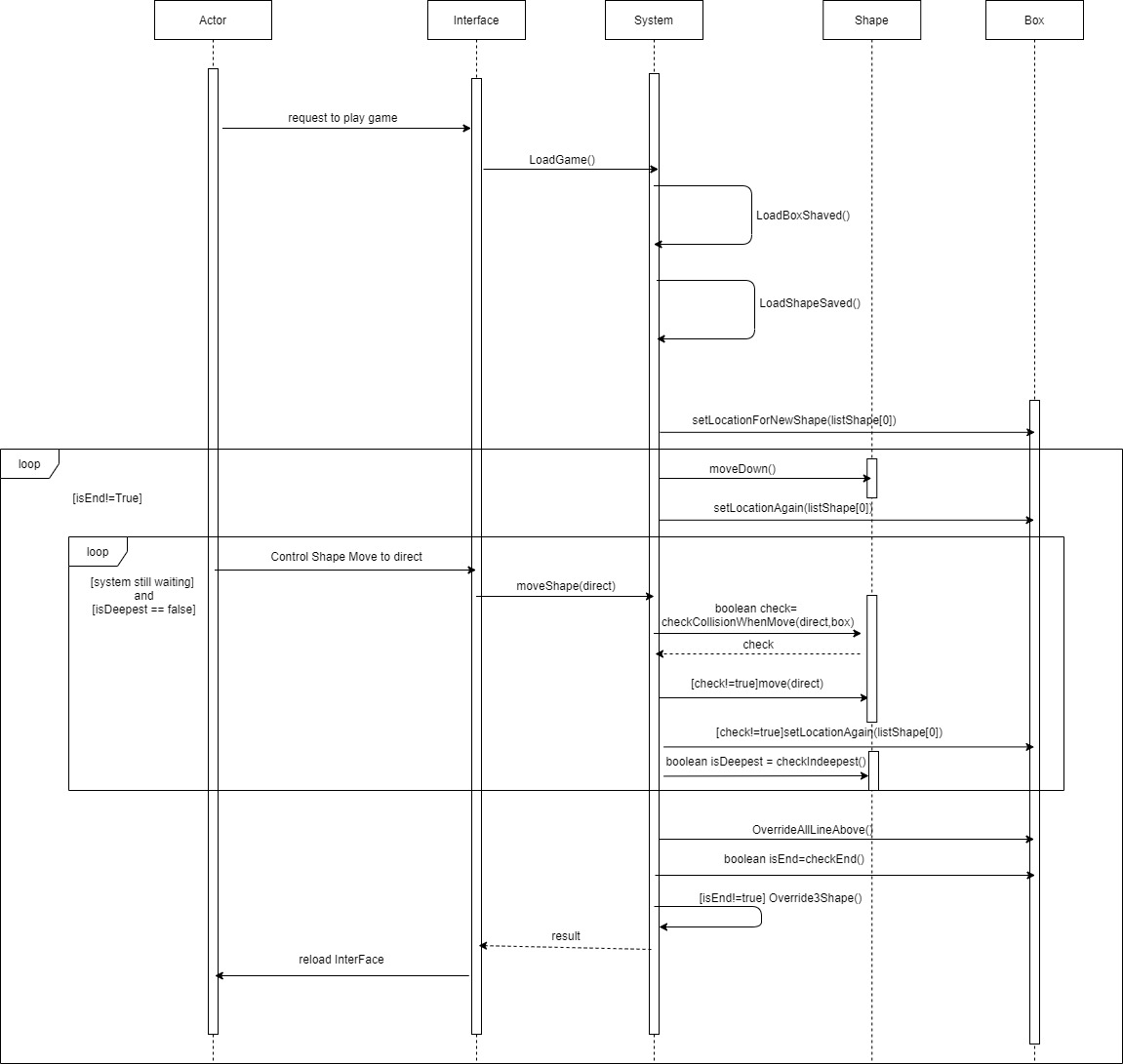


7)Sequence Diagram  
- nhờ vào Activity diagrams ở trên và focus use đã phân tích trước đó ta có thể vẽ sequence diagrams nhằm biết rõ những phương thước mà các chức năng sẽ sử dụng.

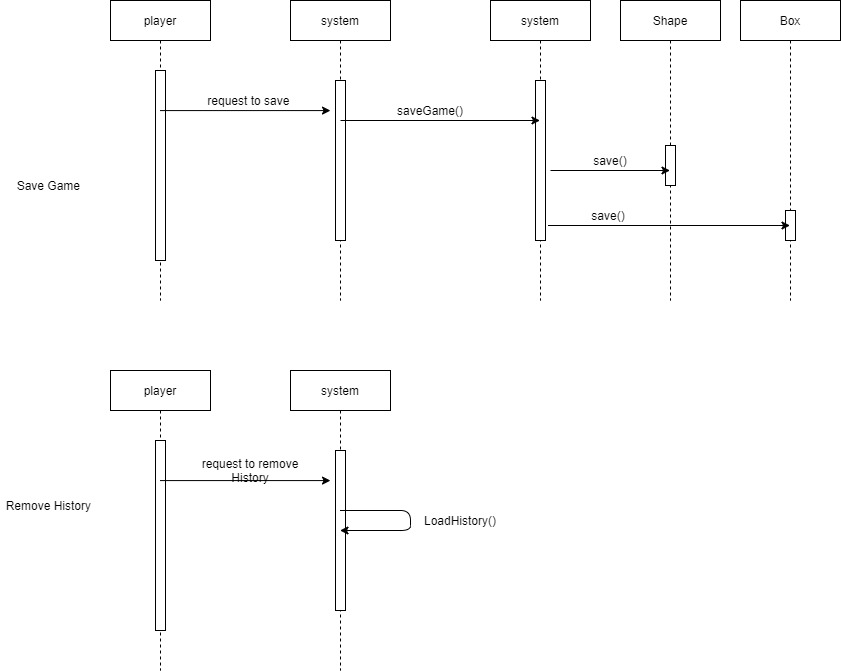
7.1)Play Game



7.2 load game

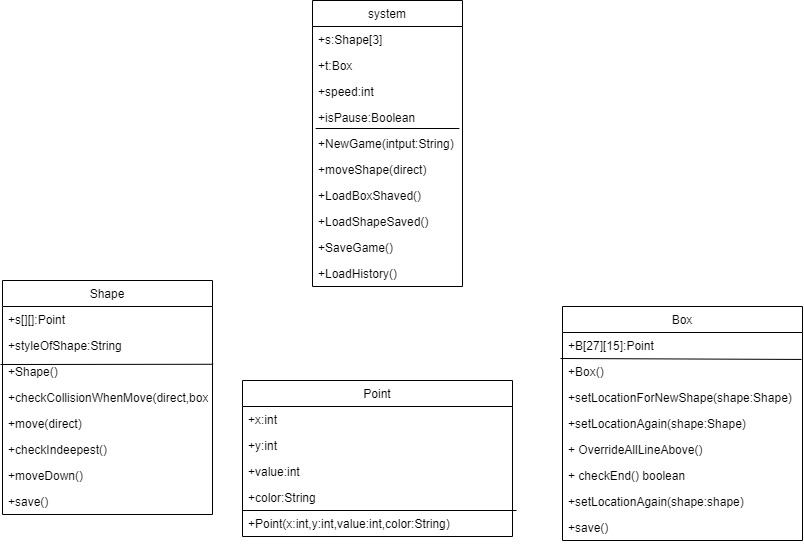


7.3 save Game & remove History



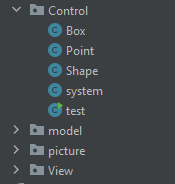
7.4 2 chức năng pause game và reload game là những chức năng vô cùng nhỏ nên ta ko cần đến sequence diagram.

8)Complete class diagram  
-từ những thứ đã phân tích trong sequence kết hợp với Sơ đồ lớp chưa hoàn chỉnh ta có được sơ đồ lớp hoàn chỉnh như sau.



Phần 2.  
1) Implement

-Sử dụng mô mình M-V-C để cài đặt game tetris



Lớp System,Box,Shape,Point được cài đặt như sau:



