**playMatch()**

1. Ciclo fino a che non ottengo exception
2. refillClouds()
3. Ciclo sulla fase di pianificazione per ogni player()

* Chiamo chooseCard
* Riceve la Card
* Effettua la modifica nel match(playCard())

1. Ciclo sulla fase di azione per ogni player:

* Per 3 chiamo moveStudent() sulla RemoteView
* Il controller riceve la pedina
* la sposta nel punto scelto dall'utente
* checkAndMoveMaster sul match
* (Gli chiede se vuole giocare una figureCard
* Il controller riceve una scelta
* Se sì gli chiede quale carta vuole giocare
* Riceve la figure card su cui chiama playFigureCard() di ExpertMatch)
* Chiede di muovere madre natura alla RemoteView
* Il controller riceve la choice
* MoveMotherNature() in match
* showClouds() in RemoteView
* Ritorna la cloud scelta
* moveFromCloudToEntrance() in match

**LIFECYCLE**

1. Model inizia il turno e chiama refillClouds(), inviando una notify del player \*NoMoreStudentsInBagException, booleano endTurn\*
2. Le view fanno l'update se sono quelle del player ricevuto, chiedono l'id della carta e fanno notify
3. Il controller svolge l'update chiamando un metodo di match per controllare che la carta non sia stata già giocata. In seguito, chiama un metodo che decide chi sarà il prossimo giocatore, il quale fa la notify
4. Ripete 2 e 3 finchè non giocano tutti i player una carta
5. Viene settato il nuovo ordine dei giocatori in base alle carte che hanno giocato
6. currentPlayer riparte dall'inizio della lista settata e chiama la notify segnalando che siamo nella fase di azione
7. La view mostra lo stato del gioco, viene selezionato uno studente e dove lo si vuole mettere per 3 volte
8. Notify al controller, che chiama moveStudentsFromEntranceToDR o moveStudentsFromEntranceToIsland per ognuno dei tre studenti a seconda della scelta e chiama checkAndMoveMaster()
9. Notify alla view, che chiede il numero di spostamenti di madre natura
10. notify al controller che chiama moveMotherNature(), il quale controlla se il dato in ingresso è valido, se valido vai al punto 11, altrimenti al 9
11. chiama checkDashboardWithMoreInfluence(), se torna #IDK chiama anche il checkNearIsland() \*Può ritornare che sono finite le torri indicando il player vincente(NegativeNumberOfTowerException)\*
12. Notify alla view che mostra il nuovo stato del gioco e le nuvole
13. Sceglie una nuvola e notify al controller
14. Il controller chiama moveFromCloudToEntrance(), in caso di errore nell'inserimento del numero della nuvola torna al punto 12
15. Passa al prossimo giocatore, se esiste torna al punto 9, altrimenti il ciclo termina, currentPlayer torna al primo della lista e si ritorna al punto

**CHOICES**

Messaggi dal match alla View:

* Figure cards: la figure card che è stata scelta dall'utente
* Fase di pianificazione -> mostrare le carte al proprietario per sceglierne una sola: devo passare una copia delle carte per permettere alla View di mostrarle e di far scegliere al giocatore
* Fase di azione, scelta delle pedine che vengono mosse dall'Entrance: tutto il modello(MatchView, una classe che maschera il Match), di modo che possa scegliere con tutti i dati a disposizione e un numero o una stringa che identifichi il momento della fase dazione("entrance choice" ad esempio)
* Fase di azione, selezione della nuvola: la MatchView per aggiornare il modello presente sulle View e un'altra stringa(Cloud choice ad esempio)

Dalla View al Controller:

* Figure cards: la figure card giocata, e poi a seconda della figure card:
  + Figure cards with students: studenti scelti presenti sulla carta, e poi:  
    - Jester: studenti da scambiare nell'ingresso

- Merchant: num isola su cui posizionare lo studente

-Princess: //

* Centaur: //
* Knight: //
* Fungaiolo: colore da bloccare in gioco
* Sciura: num isola da bloccare in gioco
* Postino://
* Fase di pianificazione: carta scelta tra quelle nel deck
* Fase di azione, scelta delle pedine che vengono mosse dall'Entrance: gli studenti scelti dall'utente e dove vuole spostare ognuno di essi(dr oppure isole e che isola)
* Fase di azione, selezione della nuvola: numero di nuvola scelto dall'utente