

PROTOCOL DESIGN

SOMMARIO

2 - Bean Messaggio Client —> Server

3 - Bean Messaggio Client <— Server

4 - Initial Setup

5 - All Macroactions

5 - Buy From Market

7 - Activate Production

8 - Buy Dev Card

9 - Activate Leader

9 - Discard Leader

9 - Move One resource

9 - Switch Resource Slots

9 - End Turn

10 - List of updates

BEAN MESSAGGIO CLIENT ---> SERVER:

```
public class Command {
    String cmd;
    Int numOfPlayers;
    String username;
    Int chosenLeader1;
    Int chosenLeader2;
    String chosenResource1;
    String chosenResource2;
    Int marketPosition;
    Int shields;
    Int stones;
    Int servants;
    Int coins;
    String resourceType;    {"shield", "stone", "coin", "servant"}
    Int slotNumber;
    Int fromSlotNumber;
    Int toSlotNumber;
    Boolean slot1Activation;
    Boolean slot1Activation;
    Boolean slot3Activation;
    Boolean baseProductionActivation;
    String baseInputResource1;
    String baseInputResource2;
    String baseOutputResource;
    Boolean leader1SlotActivation;
    Int leader1Code;
    String leader1ConvertedResource;
    Boolean leader2SlotActivation;
    Int leader2Code;
    String leader2ConvertedResource;
    Int chestCoins;
    Int chestStones;
    Int chestShields;
    Int chestServants;
    Int storageCoins;
    Int storageStones;
    Int storageShields;
    Int storageServants;
    Char devCardColour;
    Int devCardLevel;
    Int leaderCode;
}
```

BEAN MESSAGGIO CLIENT <--- SERVER:

```
public class Response {
    String cmd;
    Boolean commandWasCorrect;
    String resp;
    Int leader1Code;
    Int leader2Code;
    Int numOfInitialResources;
    Int jolly;
    Int stones;
    Int shields;
    Int coins;
    Int servants;
    Int newTotalVictoryPoints;
    Int [] newPlayersPositions = new int[4];
    Int newBlackCrossPosition;
    Boolean[] newActiveFirstPapalFavourCard = new boolean[4];
    Boolean[] newActiveSecondPapalFavourCard = new boolean[4];
    Boolean[] newActiveThirdPapalFavourCard = new boolean[4];
    Int newGreen1
    int newGreen2
    int newGreen3
    Int newPurple1
    Int newPurple2
    Int newPurple3
    Int newBlue1
    int newBlue2
    Int newBlue3
    Int newYellow1
    int newYellow2
    Int newYellow3
    String[] newFirstMarketRow = new String[4];
    String[] newSecondMarketRow = new String[4];
    String[] newThirdMarketRow = new String[4];
    String newExtraMarble
    String playerUsername;
    String[] playerUsernames = new String[4];
    String newResourceTypeOfSlot1
    String newResourceTypeOfSlot2
    String newResourceTypeOfSlot3
    Int newQuantityOfSlot1
    Int newQuantityOfSlot2
    Int newQuantityOfSlot3
    String newResourceTypeOfLeaderSlot1
    String newResourceTypeOfLeaderSlot2
    Int newQuantityOfLeaderSlot1
    Int newQuantityOfLeaderSlot2
    Boolean leader1Active;
    Boolean leader2Active;
    Int newCoinsQuantity
    Int newStonesQuantity
    Int newShieldsQuantity
    Int newServantsQuantity
    Int newCurrentPlayer
    Int leaderCardsDrawn[] = new int[4]
    Int lastActionCardUsedCode;
    String targetResources[] = new String[2]
}
```

SOLO PER IL SETUP INIZIALE SONO I THREAD DEL SERVER A COMANDARE L'EVOLUZIONE DEI MESSAGGI

INITIAL SETUP: (if the input isn't correct the server simply asks again the same thing)

(only if game == null)

Cmd = defineNumberOfPlayers

Resp = null or custom message

<-----
numOfPlayers
----->

Cmd = insertUsername

resp = null or "username already exists"

<-----
username
----->

Cmd = sorryGameAlreadyFull

resp = custom message

<-----
Close connection on server and stop the client

cmd = leaderDistribution

leaderCardsDrawn[]

<-----
ChosenLeader1
ChosenLeader2
----->

cmd = giveInitialResources

numOfInitialResources

<-----
ChosenResource1
ChosenResource2
----->

A fine del setup di tutti i giocatori: (solo il thread collegato giocatore con turn order 1 manda gli update a tutti)

Cmd = setupUpdate

Cmd = faithTrackUpdate

Poi 1 cmd = storageUpdate per ogni player del game

Poi 1 cmd = leaderCardsUpdate per ogni player del game

<-----

gameStart

<—————

... Poi i thread del server si mettono in attesa di comandi del client a cui rispondere
A questo punto i thread del server sono tutti passivi alle azioni dei client

ALL MACROACTIONS:

cmd = buyFromMarket
activateProduction
buyDevCard
activateLeader
discardLeader
placeResourceInSlot
discardResource
moveOneResourc
switchResourceSlots
endPlacing
chosenResourcesToPay
chosenSlotNumberForDevCard
endTurn

—————>

MORE IN DETAIL:

BUY FROM MARKET

Cmd = buyFromMarket
MarketPosition

—————>

commandWasCorrect
jolly
coins
stones
shields
servants
targetResources (solo se jolly != 0)

<—————

cmd = marketUpdate
cmd = faithTrackUpdate (only if faith bought)

<—————

Cmd = chosenResourcesToBuy
Coins
Stones
Shields
Servants

—————>

commandWasCorrect
coins

stones
shields
servants
(jolly only if bad request)

<-----

Cmd = placeResourceInSlot
ResourceType
SlotNumber

----->

commandWasCorrect
coins
stones
shields
servants

<-----

cmd = storageUpdate

<-----

Cmd = discardResource
ResourceType

----->

commandWasCorrect
coins
stones
shields
servants

<-----

cmd = faithTrackUpdate

<-----

Cmd =moveOneResource
fromSlotNumber
toSlotNumber

----->

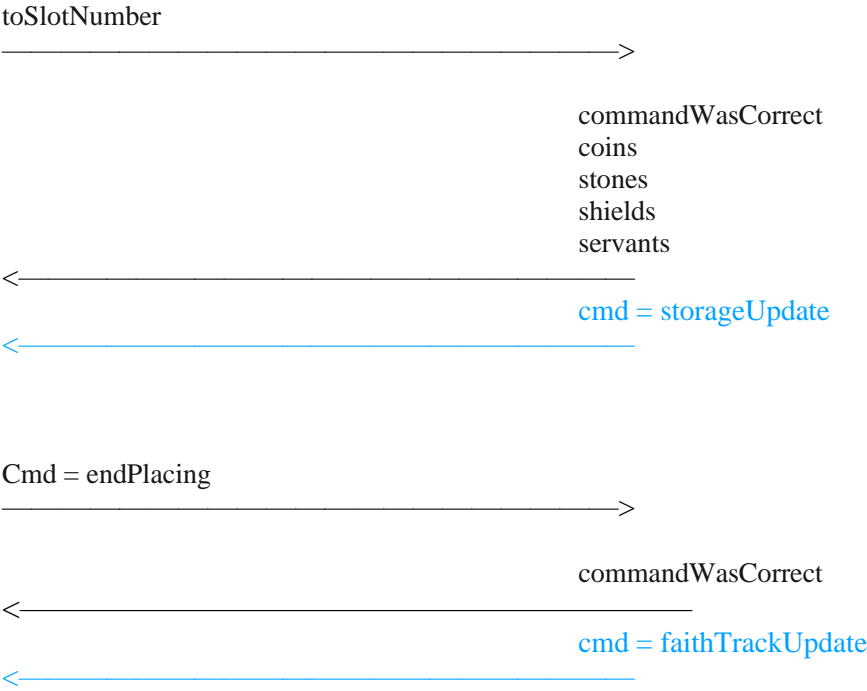
commandWasCorrect
coins
stones
shields
servants

<-----

cmd = storageUpdate

<-----

Cmd =switchResourceSlots
fromSlotNumber



ACTIVATE PRODUCTION

Cmd = activateProduction

Slot1Activation

Slot2Activation

Slot3Activation

BaseProductionActivation

BaseInputResource1

BaseInputResource2

BaseOutputResource

LeaderSlot1Activation

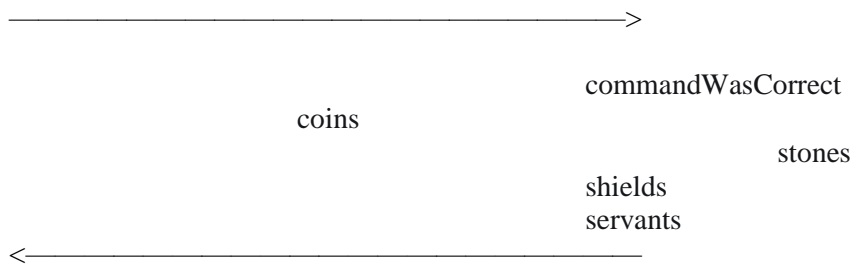
Leader1Code

Leader1ConvertedResource

LeaderSlot2Activation

Leader2Code

leader2ConvertedResource



Cmd = chosenResourcesToPayForProduction

ChestCoins

ChestStones

ChestShields

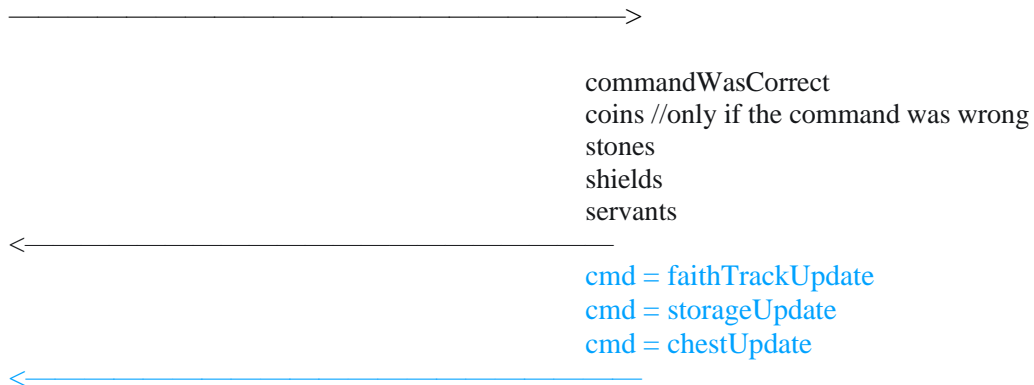
ChestServants

StorageCoins

StorageStones

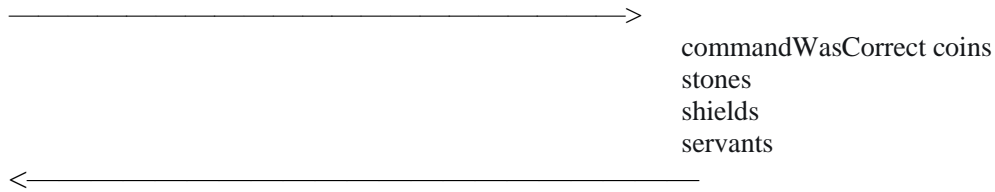
StorageShields

StorageServants

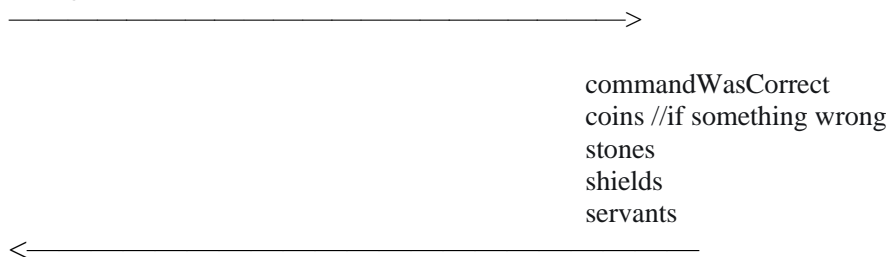


BUY DEV CARD

Cmd = buyDevCard
DevCardColour
DevCardLevel



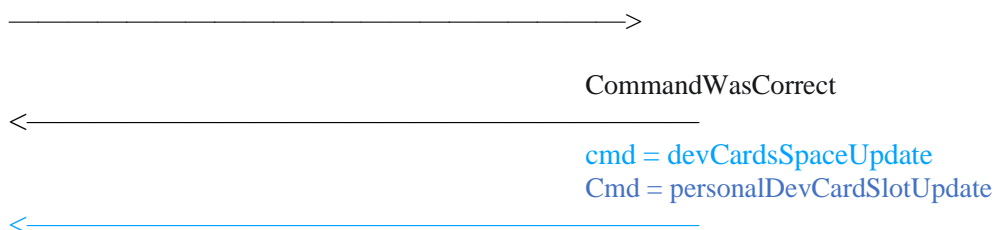
Cmd = chosenResourcesToPayForDevCard
ChestCoins
ChestStones
ChestShields
ChestServants
StorageCoins
StorageStones
StorageShields
StorageServants



cmd = chestUpdate
cmd = storageUpdate

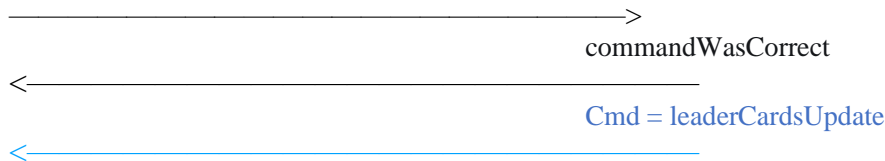


Cmd = chosenSlotNumberForDevCard
SlotNumber



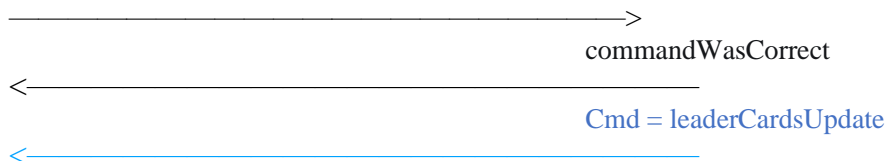
ACTIVATE LEADER

Cmd = activateLeader
leaderCode



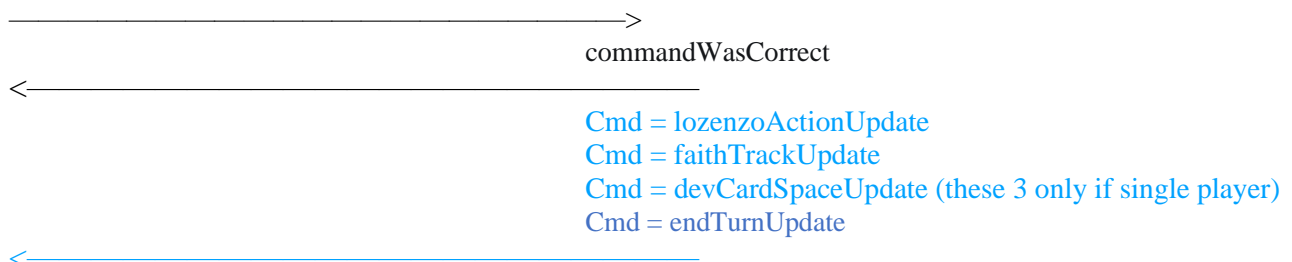
DISCARD LEADER

Cmd = discardLeader
leaderCode



END TURN

Cmd = endTurn



LIST OF UPDATES

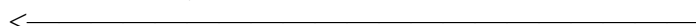
Cmd= setupUpdate
PlayerUsernames[]



Cmd = leaderCardsUpdate
PlayerUsername
Leader1Code; //if == 0 vuol dire che è stato scartato Leader1Active;
Leader1Active;
Leader2code; //if == 0 vuol dire che è stato scartato Leader2Active;
Leader2Active;



Cmd = totalvictorypointsUpdate
newTotalVictoryPoints



Cmd = fathTrackUpdate
newPlayersPositions[]

newBlackCrossPosition
newActiveFirstPapalFavourCard[]
newActiveSecondPapalFavourCard[]
newActiveThirdPapalFavourCard[]

<

cmd = devCardSpaceUpdate NewGreen1
NewGreen2
NewGreen3
NewPurple1
NewPurple2
NewPurple3
NewBlue1
NewBlue2
NewBlue3
NewYellow1
NewYellow2
NewYellow3

<

Cmd = marketUpdate
NewFirstMarketRow[]
NewSecondMarketRow[]
NewThirdMarketRow[]
NewExtraMarble

<

Cmd = StorageUpdate
playerUsername
NewResourceTypeOfSlot1
NewResourceTypeOfSlot2
NewResourceTypeOfSlot3
NewQuantityOfSlot1
NewQuantityOfSlot2
NewQuantityOfSlot3
NewResourceTypeOfLeaderSlot1
NewResourceTypeOfLeaderSlot2
NewQuantityOfLeaderSlot1
NewQuantityOfLeaderSlot2

<

Cmd = chestUpdate
playerUsername
NewCoinsQuantity
NewStonesQuantity
NewShieldsQuantity
NewServantsQuantity

<

```
Cmd = personalDevCardSlotUpdate  
playerUsername  
newDevCardCode  
stackSlotNumberToPlace
```

<—————

```
Cmd = lorenzoActionUpdate  
lastActionCardUsedCode
```

<—————

```
Cmd = endTurnUpdate  
newCurrentPlayer
```

<—————

```
Cmd = printOutUpdateMessage    { this is an extra message we can use to print something to all  
players screens }  
Resp
```

<—————