JAGSAT project

Design Document

Juan Pedro Bolívar Puente
Aksel Junkkila
Guillem Medina
Sarah Lindstrom
Alberto Villegas Erce
Thomas Forss

Project Course
Åbo Akademy

December 10, 2009

Contents

1 Introduction

The aim of this document is multiple, on one hand is a way to establish communication gates between our client and us, showing what and how we plan to do as a matter of contract. On the other hand is a guide to follow by our developers, well defined steps and rules of what our client wants to be done.

In the following pages there will be found several descriptions about interfaces, use cases, rules and design of the project. Due to the nature of the project there will probably be some inconsistencies between this document and the final released product. As a game itself, its design relies on the playability which, in any case, can only be measured empirically through a process of testing and continuous elliptical modifications and refining.

TO-DO note 1 (Juan Pedro) Add description about the purpose of the document and bla bla bla. I think that, given the inconsistencies between different parts of the document and also the potential inconsistencies between this document and the final released product, we could include a disclaimer stating that game design relies on playability and that playability can be only measured empirically, therefore requiring continuous elliptical modifications and refining.

2 Game design

2.1 Introduction

This section describes the overall game design. Firstly a detailed view of the game rules, in both parts of the game. These rules describe what players will be able to do and not to do. Secondly an overview of the user interface which, in any case, is deeply detailed and illustrated in section 7. Thirdly, and finally, a detailed list of the graphics and animations that will be needed in the final version.

TO-DO note 2 (Juan Pedro) Improve this introduction. Also, some parts of this section might need some refactoring. For example, the *Graphics* part might be worth having its own section, also the *User interface* could be integrated with the *User interface* later section.

TO-DO note 3 (Alberto) Can we consider this introduction as done?

TO-DO note 4 (Juan Pedro) While reading the whole text during the refactor we have found important inconsistencies. The Axel description of the rules and interface does not match very well with my state machines and the UI diagrams. Also, some parts of it are too schematic and hard to understand that could be improved with longer descriptions and better ordering of some items. Also, the list of graphics needed for the project is too long, there are many things that are not needed (afaik) or clear. For example, we don't need unit pictures for the world map, because it much more reasonable to use numbers to represent the number of units.

2.2 World Domination

2.2.1 General rules

TO-DO note 5 (Juan Pedro) There are some misunderstandings in the rules. Maybe we can copy paste from the Wikipedia page (be careful with the license, but probably there is no problem on making this doc GFDL, actually the `.doc` version was...): `www.wikipedia.org/wiki/Risk_(game)`

- Every player has an objective, the objectives are determined by drawing cards.
- World map is divided into countries, every player gets countries on random.
- Every player gets troops and deploy them in their countries.
- The world domination game is played in turns.
- Starting player is chosen by random.
- Player turn order is clockwise.

2.2.2 During turn

- There are 3 phases in a players turn. Reinforcements, attacking and moving.
- Reinforcements can be placed within countries with a ratio of 1/3 of each country. Exception on first turn where the minimum is one unit.
- The player receives additional units based on territories and continents.
- Once the player has possession of nine territories, for every three additional territories in possession, the player gains an additional unit per turn.
- The player may also receive units if he turns in a set of unit cards. He then places the units on any of his territories.
- If the player has conquered at least one territory, he draws a unit card from the deck. Once the player has enough cards to acquire a unit the game will do so automatically.
- A set of unit cards consists of one of the following:
  - Three cards depicting the same unit (eg. all three cards have infantry pictures)
  - Three cards showing one of each type of Risk unit (infantry, barrier, cannon).
- A country must always have one unit in it.

- A unit can only move to a country next to the country it is in. *Optional.*
- A player can attack nearby countries from the country he owns.
- A player must have more than one unit in a country to be able to attack another country.
- When a player attacks he attacks with all units he has in the country he is attacking from.
- The defender defends with all units he has in the country he is defending.
- If the defender only has one unit in a country he is defending. He can not conquer the attackers country.
- A conquered country cannot attack nearby countries within the same turn it gets conquered.
- Units can only be used once every turn.
- A unit is considered used when it destroys an enemy unit or passes the line for the defending country.
- A unit is also considered used when it has been moved to another country.

2.3 Tower defense

2.3.1 General rules

- Tower defense gameplay is live.
- Tower defense battlefield is divided into 4 zones consisting of a defending zone and an middle zone for both players. This means both players have an area to defend and a middle area to maneuver in.
- The attacker and defender both get 3 different kind of units. These are cannons, infantry and barriers.
- There is an option of surrendering, the time when you can surrender is determined randomly. The timelimit for surrendering is 10 seconds. Exception: You can not surrender in the beginning.

- There is a time limit to a battle. If the winner has not been determined by that point the battle will be considered a tie and the units lost will be registered.
- There is the option to choose the speed of the battle game. *Optional*.

2.3.2 The set up

- Setting up units will be live. A player can choose when to place his units on the field.
- Before the battle begins, the defender will flip a coin. For every right coinflip he will get one more unit up to a maximum of the attackers amount of units. If the attacker has less units than the defender there will be no coinflip.
- Attacker gets infantry, cannons and barriers equaling to the amount of units he his attacking with. If the attacker is attacking with 5 units he will get 5 of each.
- The defender gets infantry, cannons and barriers equaling to the amount of units he is defending with.
- The defender also has the chance on getting extra units, this will be determined randomly. The defenders extra units will not count as units on world map.
- Cannons can be set up at defense zone only. Their placement will be random and the cannons will be stationary.
- Infantry can be set up anywhere in the players own defending area or middle area.
- Barriers can only be set up at players own middle area.

2.3.3 Gameplay

- Cannons can destroy barriers.
- Cannons can also destroy enemy cannons.
- Barriers can block infantry. A blocked infantry dies.

- Infantries are equal and destroy each other if they meet.
- Infantry can pass the defense line. An infantry passing the defense line will kill an enemy unit on the world map. Thus if enough infantry pass the line the country gets conquered.
- Infantry can bounce from battlefield edges.
- Cannonballs do not bounce but disappear if they are shot out of the battlefield area.

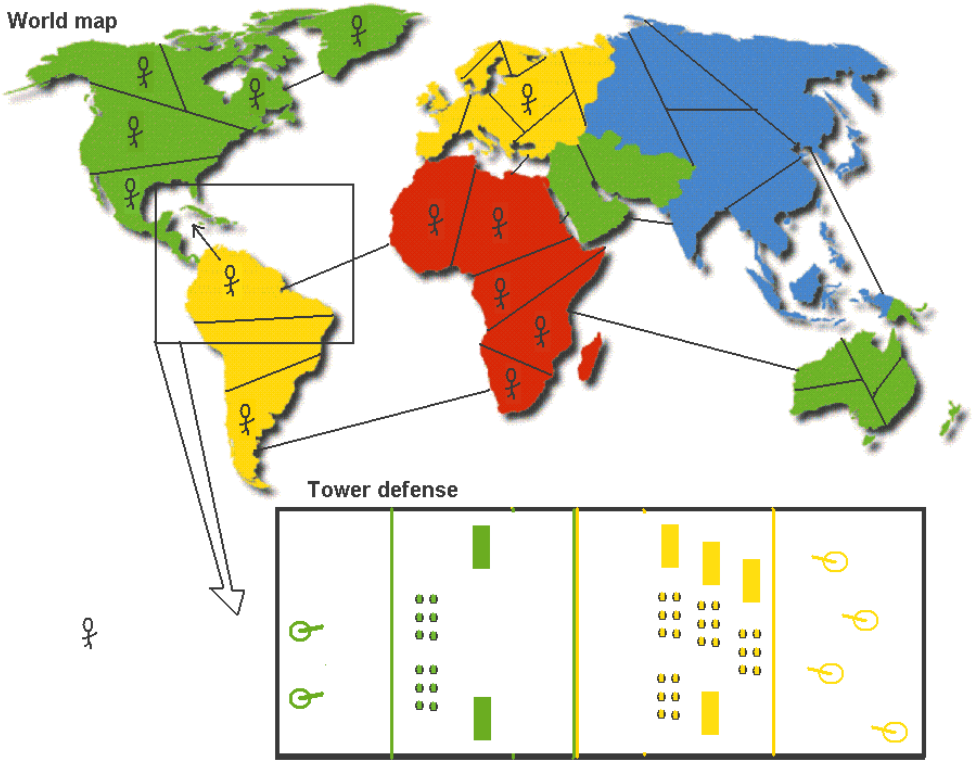


Figure 1: Mockup of the game play interface.

2.4 Interface

2.4.1 World map

- Objective cards are shown when clicking on the objective button.

- In the deployment and reinforcement phase. A player can deploy his units by clicking on a unit icon and then clicking on a country. If he wants to deploy several units to the same country he can click the icon several times and a counter will show how many units he has chosen.
- In the deployment phase a player can move units from one country to another without any limits.
- A player can move units from one country to another by clicking on a country he wants to move units from. If he wants to move several units at the same time he can do so by clicking several times on the country and a counter will show how many units he has chosen.
- Attacking a country is done by clicking on an own country and then clicking a nearby enemy country.

2.4.2 Tower defense

- A player can set up infantry anywhere on his own zones. He will do so by clicking on a infantry icon and then clicking on a spot on one of his zones. After the infantry has been deployed he steers the infantry in a direction by clicking on the screen. An arrow will point the direction.
- A player can set up barriers anywhere on his own middle zone. He will do so by clicking on the barrier icon and then clicking again on a spot that he wants to block. The barrier is now set and can not be moved again.
- Cannon placement is randomized within the defense zone. A player can choose a cannon by clicking on it. When he chooses a cannon an arrow appears, he can steer the arrow in a direction by dragging with his finger on the display. The more he drags the longer will the shot be.
- A player can alter the movement of his infantries while they are within his own middle zone. He will do so by clicking on a infantry unit and an arrow appears. He will choose the new direction by clicking on the screen again.
- A currently chosen unit will be highlighted.

- When the surrender time option is available a surrender button will appear for both players. If a player clicks on the button he will surrender. When surrender time is over the button disappears.
- Before the actual battle begins the defender flips a coin for more units. He will do so by choosing heads or tails and a randomizer will show the result.

2.5 Graphics

2.5.1 Main Intro. *optional*

2.5.2 Main screen

- Main game menu
- New game options screen
- Difficulty
- Number of players
- A base character that will work as units on main map
- Different characters to choose from *Optional*.
- Different colors for player characters
- Sound menu

2.5.3 Main Map

- Divided in areas
- Even and squared or uneven areas
- Size always same but more areas depending on difficulty. *Optional*.
- Player name whose turn it is, is shown
- Turn number shown/ Year or equivalent
- Player controlled areas shown

- Different colors for each players
- Player units shown
- Different colors on units aswell
- Animations on main map. *Optional*.
- These would be extras
- Attack screen/ animation shown when attacking. *optional*.

2.5.4 Battlefields

- Different backgrounds depending on country/area
- Forrest
- Size of battlefield always same
- Bridges
- Water / Rivers
- Rocks
- Generating random battlefields. *Optional*.
- Time limit shown
- Roles, the defender and the attacker shown
- Defending and middle zones shown

2.5.5 Units

2.5.6 Main Map Units

- Player units
- Shown in a way so that all players can see everyones units
- Battlefield Units
- Troops

- Health shown. *Optional*.
- Cannons
- Health shown. *Optional*.
- Barriers
- Health shown. *Optional*.

2.5.7 Animations

- Troop animations: *Movement, Shooting, Death*
- Cannon animations: *Movement, Shooting, Death*
- Barrier animations: *Placement (Optional), blocking infantry, Destroyed by cannon*
- Player unit movement on main map
- Area highlighting when chosen
- A players battlefield unit currently in control is shown somehow
- Highlighted, with the specific player color
- Going into battlefield animation *optional*.
- Attack animation on main map *optional*.

3 Use cases

3.1 Start Game

Priority 1

Users User, game.

Precondition None.

Flow

1. Users open game.
2. Game plays a short clip.
3. Game menu initiates.

Postcondition Program is running.

Alternative Flow 1

1. User opens game.
2. Game menu initiates.

3.2 Player settings

Priority 3

Users User, game

Precondition Game is initiated.

Flow

1. Player chooses *New Game* in menu.
2. User chooses amount of players.
3. User enter new names.

Postcondition None.

Alternative Flow 1

1. User chooses amount of players.

2. User chooses default names.

Alternative Flow 2

1. User goes back to menu.

3.3 Game settings

Priority 3

Users User, game

Precondition Amount of players set.

Flow

1. User chooses a map from the list.
2. Game starts.

Postcondition Game starts.

Alternative Flow 1

1. User goes back to edit player settings.

TO-DO note 6 (Juan Pedro) Choosing the game length might be no longer needed because we now let the user choose an arbitrary map from the list?

3.4 Sound settings

Priority 4

Users User, game

Precondition Game menu is initialized.

Flow

1. User chooses *Sound settings* from menu.
2. User changes sound settings.
3. User saves changes.

Postcondition Settings are changes.

Alternative Flow 1

1. User chooses *Sound settings* from menu.
2. User discards changes.

3.5 Player takes turn

Priority 1

Users User, game

Precondition Game has started.

Flow

1. Player places reinforcements on field.
2. Player attacks enemy countries.
   (a) Tower defence starts.
   (b) Tower defence ends.
3. Player moves troops.
4. User ends turn.

Postcondition Turn is over.

3.6 Attacker (Tower defence)

Priority 3

Users User, game.

Precondition Player attacks enemy area.

Flow

1. Amount of units in area are converted into money.
2. User chooses what units to build.
3. Player places units on field.
4. Player passes enough units onto enemy area.

5. User takes control of area.
6. All units but one moves into new area.

Postcondition None.

Alternative Flow 1

1. Player chooses what units to build.
2. Player places units on field.
3. Player runs out of money.
4. Player fails to pass enough units to enemy area.
5. Player does not take control of area.

Alternative Flow 1

1. Player chooses what units to build.
2. Player places units on field.
3. Player runs out of time.
4. Player does not take control of area.

TO-DO note 7 (Juan Pedro) This use-case does not reflect the game rules. One does not have money anymore, and the defender can actually takeover the attacker country.

3.7 Defender (Tower defense)

Priority 3

Users User, game

Precondition User attacks enemy area.

Flow

1. Amount of units are converted into money.
2. Player chooses what units to build.
3. Player places units on field.
4. Attacker fails to pass enough units.

5. User defends area.
6. User keeps control of area.

Postcondition None.

Alternative Flow 1

1. Player chooses what units to build.
2. Player passes enough units to attackers side.
3. Player keeps control of area.

Alternative Flow 2

1. Player chooses what units to build
2. Player runs out of money.
3. Attacker fails to pass enough units.
4. Player keeps control of area.

Alternative Flow 3

1. Player chooses what units to build.
2. Player runs out of time.
3. Player keeps control of area.

3.8 Player is eliminated

Priority 2

Users User, game.

Precondition Game has started.

Flow

1. Player loses last area.
2. Player is removed from turn list.
3. Losing sound is player.

Postcondition Player is out of the game.

3.9 Player wins the game

Priority 2

Users User, game

Precondition Game has started.

Flow

1. Player take control of last area.
2. All other users are removed from turn list.
3. Winning sound is played.
4. Game statistics are showed.

Postcondition Game is over.

Alternative Flow 1

3.10 Save game

Priority 4

Users User, game.

Precondition Game is active.

Flow

1. User choose *Save and quit game*.
2. User specifies save game.

Postcondition Game ends.

3.11 Load game

Priority 4

Users User, game.

Precondition Program is open.

Flow

1. Users chooses *Load game*.
2. User specifies load game.

Postcondition Game re-initiated.

3.12 Game start (World domination)

Priority 1

Users User, game.

Precondition None.

Flow

1. Each player is given an objective.
2. World map is divided on random.
3. Each player deploys troops depending on country size.
4. Starting player is chosen by random.
5. Players take turn in clockwise order.

Postcondition None.

3.13 Reinforcements (World domination)

Priority 2

Users User, game.

Precondition Player takes turn.

Flow

1. Player receives 1 reinforcements for each 3 countries he owns.
2. Player receives reinforcements for continents controlled.
3. Player places reinforcements on the game board.

Postcondition None.

Alternative Flow 1

1. Player has 3 conquer cards of the same type.
2. Player receives 1 troop at next reinforcement.

Alternative Flow 2

1. Player has 3 conquer cards of different type.
2. Player receives 2 troop at next reinforcement.

TO-DO note 8 (Juan Pedro) The reinforcements are actually given in a different way. The cards are exchanged in any moment during the user reinforcement phase (at user will, he can keep them for later use also) and the values that he gets for it are different.

3.14 Attack (World domination)

Priority 2

Users User, game.

Precondition Player controls at least 2 units in area and player takes turn.

Flow

1. Player marks a country that has at least two unused units.
2. Player selects neighbouring enemy country to attack.
3. Tower defence is initiated.

Postcondition None.

3.15 Movement (World domination)

Priority 2

Users User, game.

Precondition Player takes turn.

Flow

1. Player selects number δ of troops in a country that have not moved this turn.
2. Player selects neighbouring country controlled by the same player.
3. Troops move.

Postcondition δ troops are moved from the source country to the target.

3.16 Conquering country (World domination)

Priority 2

Users User, game.

Precondition Player takes turn.

Flow

1. Attacking player wins tower defence battle.
2. All units except one move into conquered country.
3. Player draws conquer card.

Postcondition None.

Alternative Flow 1

1. Defending player wins tower defence battle (no surviving attackers).
2. One unit move into attackers country.
3. Player draws conquer card.

TO-DO note 9 (Juan Pedro) The defender should not draw a conquer card AFAIK. Also, the attacker gets the amount of troops used (that crossed the border) into the other country if he wins. In any case the troops that crossed the border are marked as used.

3.17 Retreat (Tower defence)

Priority 3

Users User, game.

Precondition Tower defence is initiated.

Flow

1. Attacking player uses retreat option.
2. Tower defence is ended.
3. Surviving attacking units go back to attackers country.

Postcondition None.

Alternative Flow 1

1. Defending player uses retreat option.
2. Tower defence is ended.
3. Attacking player conquers country.
4. Surviving defending units move into closes country (random if several).

3.18 Build (Tower defence)

Priority 3

Users User, game.

Precondition Tower defence initiated.

Flow

1. Player selects troops.
2. Player selects location.
3. Player selects angle.
4. Player selects speed.

Postcondition None.

Alternative Flow 1

1. Player selects barrier.
2. Player selects location.
3. Player selects angle.

3.19 Coin flip (Tower defence)

Priority 4

Users User, game.

Precondition Tower defence is initiated. Attacker has more units than defender.

Flow

1. Defending player flips coin.
2. Coin flip successful.
3. Virtual unit (only valid during this tower defence battle) added to defender.
4. Repeat if attacker has more units.

Postcondition None.

Alternative Flow 1

1. Defending player flips coin.
2. Coin flip fails. No more coin flips.

3.20 Collision (Tower defence)

Priority 3

Users User, game

Precondition Tower defence is initiated.

Flow

1. Two troops from different team collide.
2. Both troops die.

Postcondition None.

Alternative Flow 1

1. Troop collides with barrier.
2. Troop bounces away at same angle as it entered.

TO-DO note 10 (Juan Pedro) I think that we agreed on Axel suggestion about troops not bouncing but just dying on collision with a barrier (can represent the unit being killed by the archers or something).

3.21 Shoot cannon (Tower defence)

Priority 3

Users User, game.

Precondition Tower defence is initiated.

Flow

1. Player selects cannon.
2. Player selects angle.
3. Player selects speed.
4. Player presses shoot.
5. Player hits barrier or cannon.
6. Barrier or cannon is destroyed.

Postcondition None.

Alternative Flow 1

1. Player selects cannon.
2. Player selects angle.
3. Player selects speed.
4. Player presses shoot.
5. Player misses.

3.22 Change direction (Tower defence)

Priority 4

Users User, game.

Precondition Tower defence is initiated.

Flow

1. Player selects a troop on his side of the field.
2. Player changes direction.
3. Unit changes direction.

Postcondition None.

3.23 Used unit (Tower defence)

Priority 3

Users User, game.

Precondition Tower defence is initiated.

Flow

1. A unit passes enemy defences in tower defence.
2. The unit is marked as used.
3. A unit is deducted form the opponents country units and considered *dead*.

Postcondition None.

4 Class design

4.1 Introduction

This section models the static design of the project using class diagrams that provide a bird-sight view of the code structure based on classes and their relationships.

4.2 Global diagram

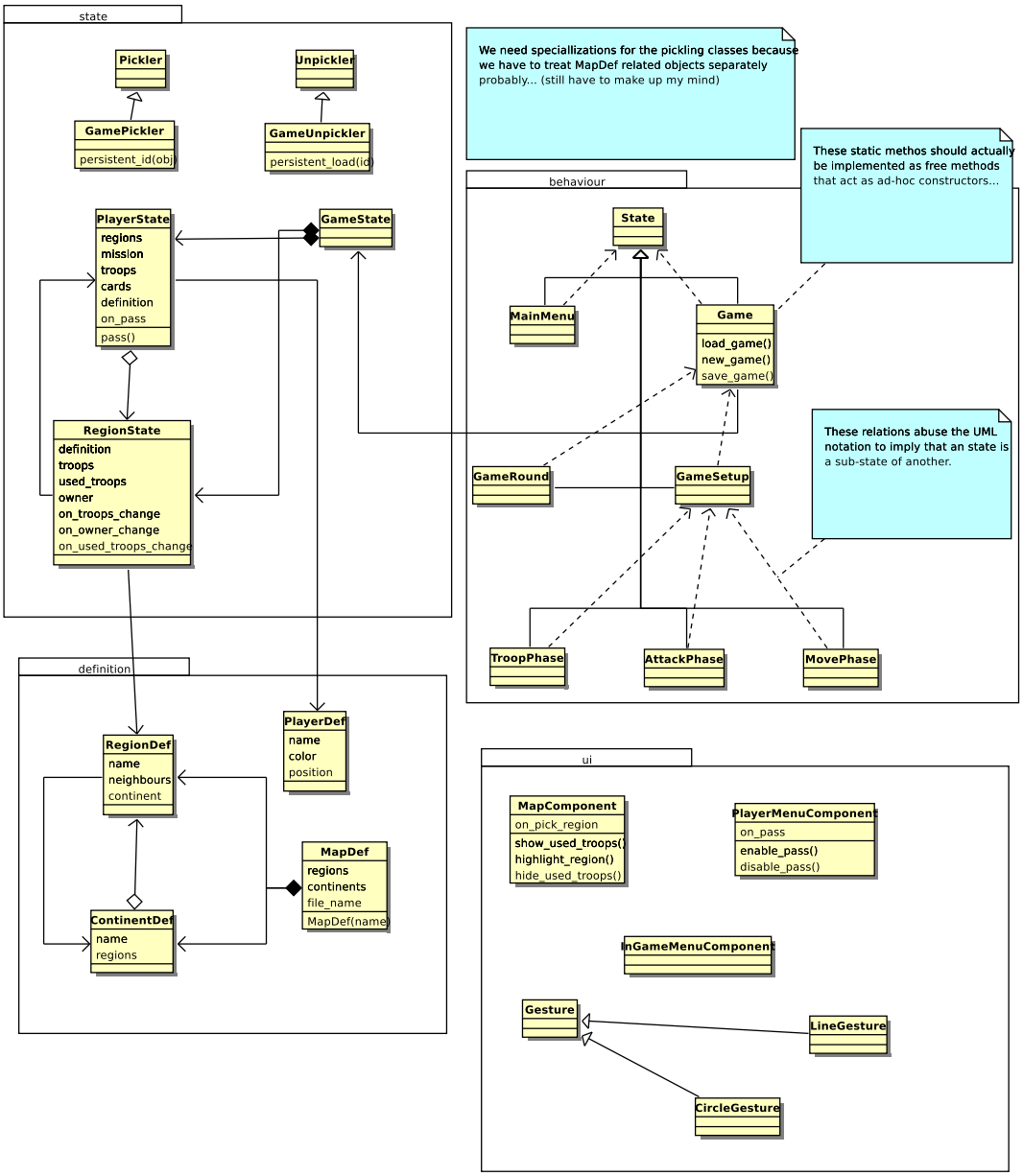


Figure 2: Class diagram.

TO-DO note 11 (Juan Pedro) This diagram is not finished. But I must admit that I consider a bit pointless finishing it right now... Also, splitting it into multiple diagrams can be a good idea. Any suggestions?

5 State machines

TO-DO note 12 Maybe some further description and connection to the use cases in the state diagrams would be good...

5.1 Introduction

This section models the dynamic behaviour of the program using class diagrams that model main interaction through events and states and the actions associated to them.

5.2 Global state machine

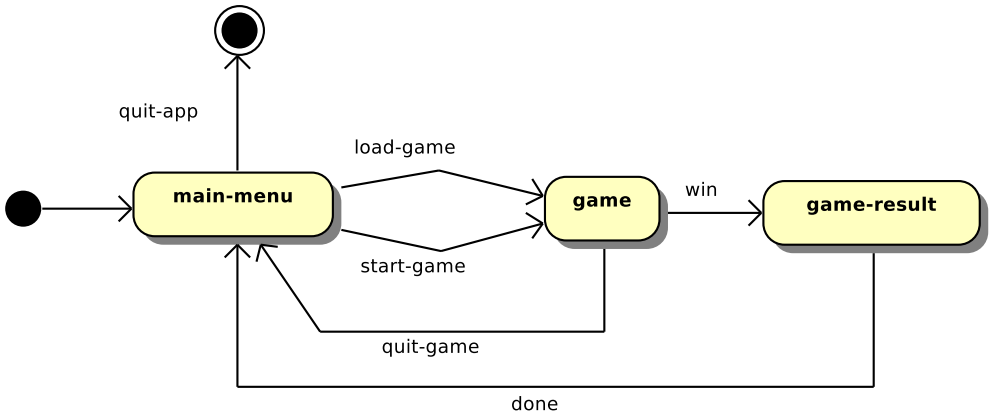


Figure 3: Global state machine.

5.3 `game` state machine.

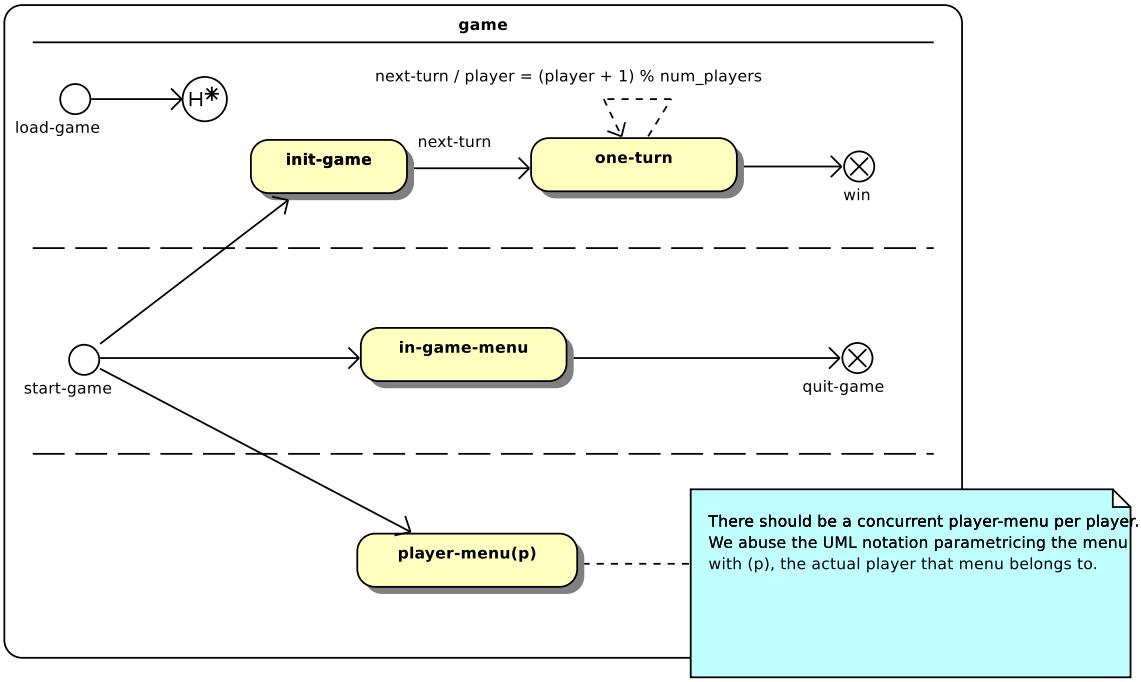


Figure 4: `game` state machine.

5.4 `init-game` state machine.

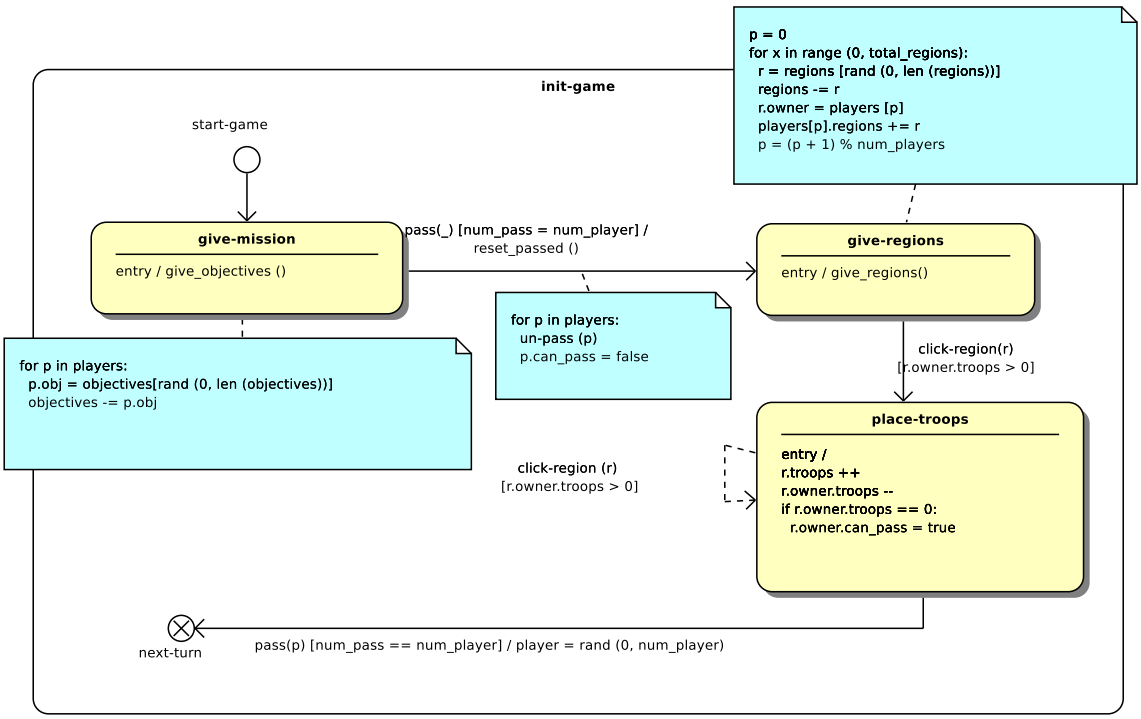


Figure 5: `init-game` state machine.

5.5 `in-game-menu` state machine.

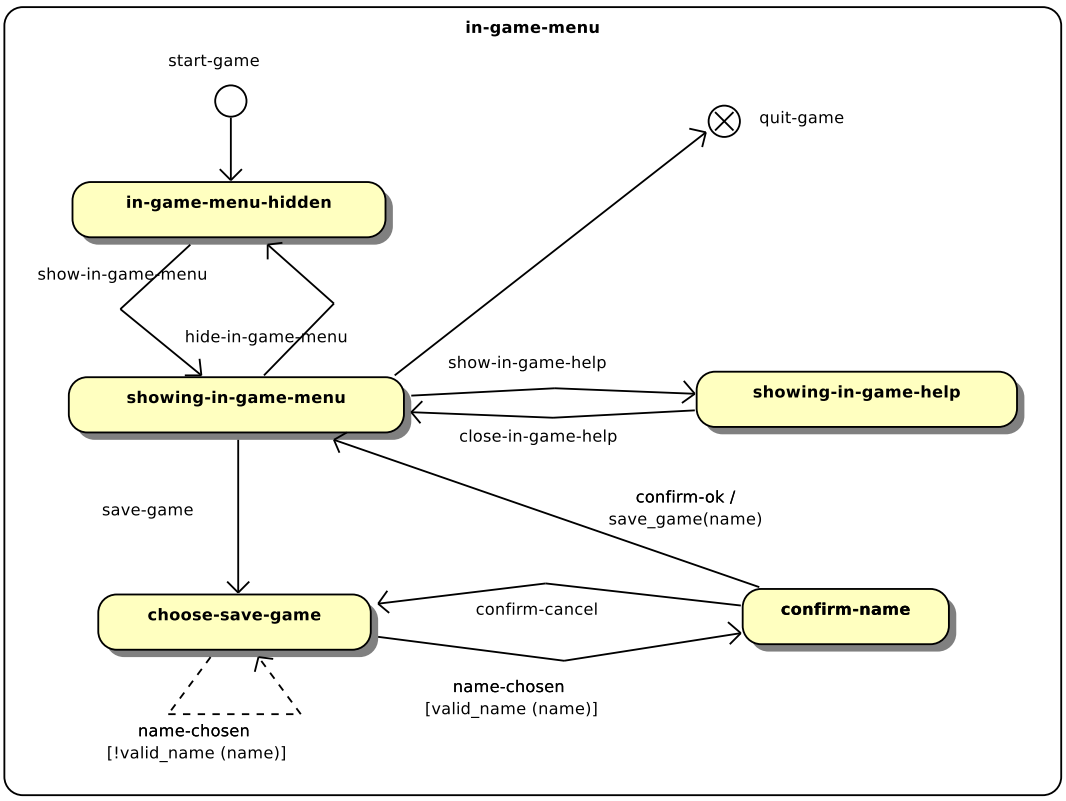


Figure 6: `in-game-menu` state machine.

5.6 `player-menu` state machine

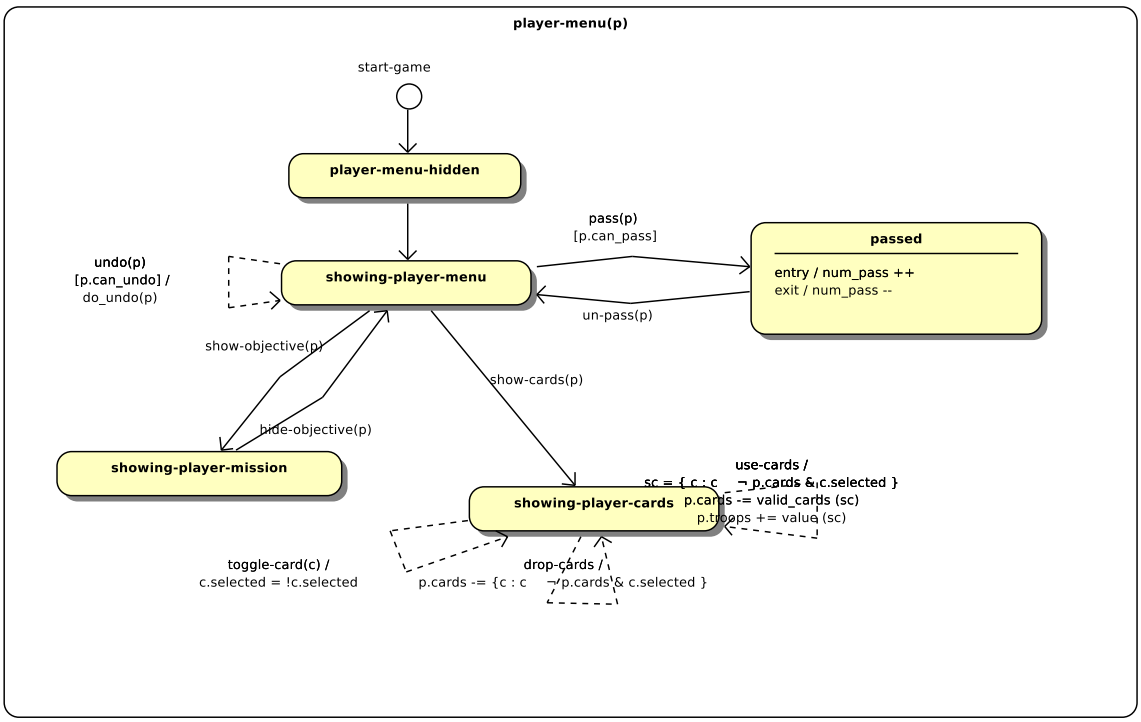


Figure 7: Global state machine.

5.7 `one-turn` state machine

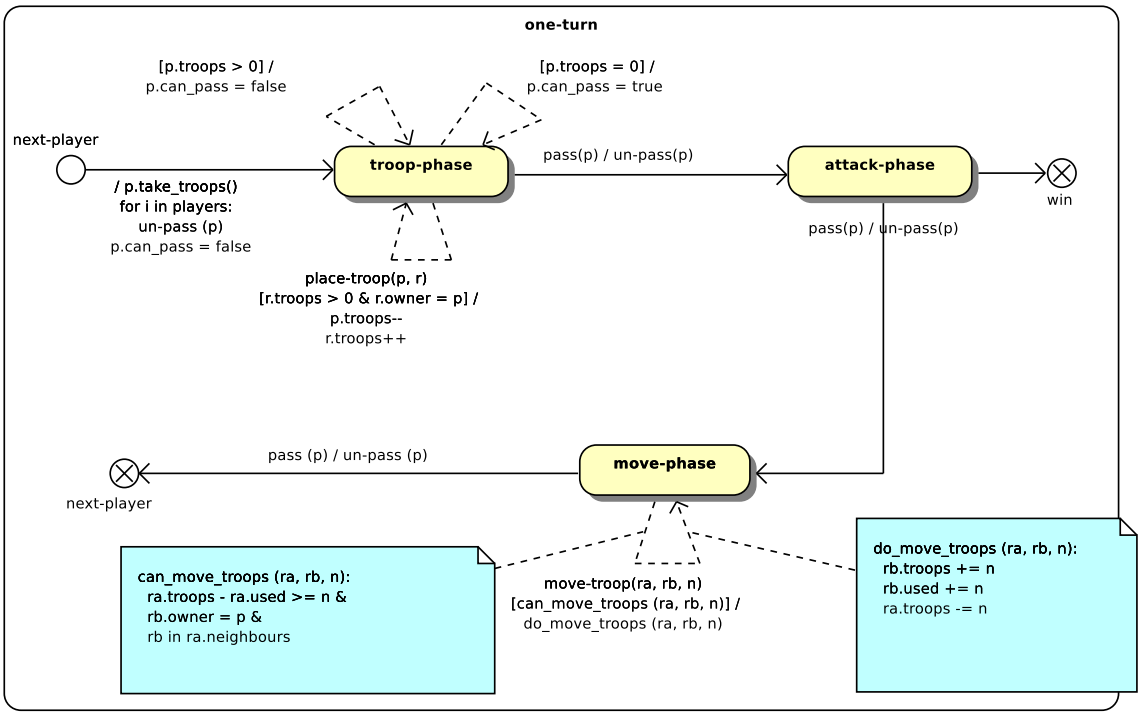


Figure 8: `one-turn` state machine.

5.8 `attack-phase` state machine

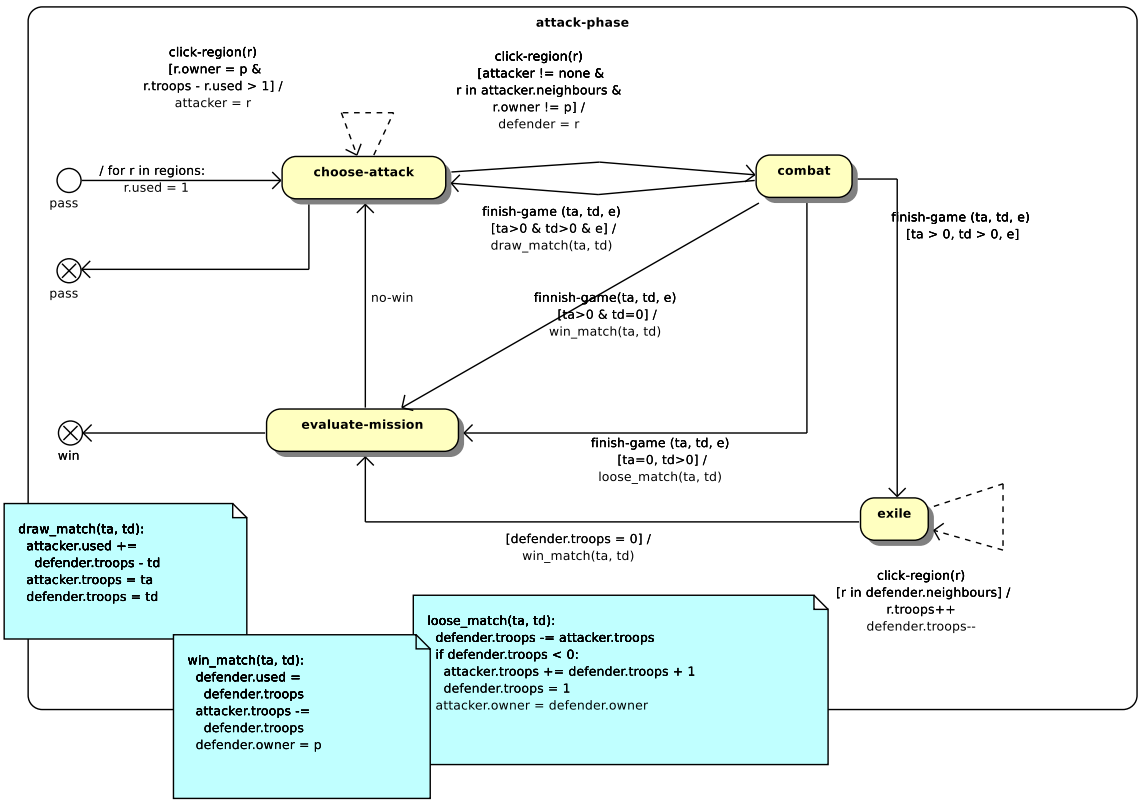


Figure 9: `attack-phase` state machine.

6 Persistence

6.1 Introduction

This section describes the different storage mechanism used by the system, as the map (boards) where the game takes place, the status for loading and loading games and the storage of player settings and profiles.

6.2 Map

In order to make the game more flexible and interesting, the players will be able to create and use new maps for them games, than can be larger or smaller (levering in this way the length of the game) or have different and challenging topologies or resemble fantastic or real places. The maps will be defined by a simple XML (eXtensible Markup Language) file. The file is defined using DTD (Document Type definition) to determine the grammar that these must follow. The semantics of the different tags and attributes that one may use in these files is encoded in comments in the DTD file that is now shown.

6.3 Map file definition `map-file.dtd`

```
1  <!--
2
3     This is the DTD definition file for the map files in the Jagsat
4     project for the Project Course in Âbo Akademy.
5
6     (c) Juan Pedro íBolvar Puente 2009
7
8  -->
9
10 <!--
11
12    Root element. The attribute 'bg' is the relative path to the
13    background image and is compulsory
14
15 -->
16 <!ELEMENT map (meta?,(continent|link)*)>
17 <!ATTLIST map
18           bg CDATA #REQUIRED>
19
```

```
<!--

   Meta-information about the map that can be used in the map selector
   or just for credit purposes.

-->
<!ELEMENT meta (author?, description?, title?)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT title (#PCDATA)>

<!--

   A continent is a group of regions. It may contain a unique 'name'
   attribute that gives a referentiable and showable name for
   it. Also, it may contain a 'troops' parameter that indicates the
   number of reinforcements that owning this continent rewards the
   player.

-->
<!ELEMENT continent (region*)>
<!ATTLIST continent
          name   CDATA #IMPLIED
          troops CDATA "0">

<!--

   A region is a country or a delimited area that the player can
   own. Its click area can be defined by either a circle or a
   polygon. The optional 'name' parameter can be used to give it a
   unique name to reference it in the linkions.

-->
<!ELEMENT region (point|circle|polygon)>
<!ATTLIST region
          name CDATA #IMPLIED>

<!--

   A circle is defined by a point and a radius. The 'radius' is an
   attribute while the center must be specified using a point.

-->
<!ELEMENT circle (point)>
<!ATTLIST circle
          radius CDATA #IMPLIED>

<!--

```

```
69     A poligon defined by the sequence of points that determine its
70     frontier polyline.
71
72  -->
73  <!ELEMENT polygon (point*)>
74
75  <!--
76
77     A point must contain an 'x' and 'y' attribute defining its absolute
78     coordinates.
79
80  -->
81  <!ELEMENT point EMPTY>
82  <!ATTLIST point
83            x CDATA "0"
84            y CDATA "0">
85
86  <!--
87
88     Use 'link' to link several regions so they can be considered
89     nerby countries and units can move between them. Note that all the
90     linkions are bi-directional. It contains a sequence of
91     nodes. Depending on the value of the 'type' attribute this will
92     behave:
93
94     - line: The given nodes are going to be sequentially. For example,
95     if the nodes named 'a', 'b', 'c' and 'd' are given, the following
96     set of linkions will be created: 'a-b,␣b-c,␣c-d'
97
98     - circle: Same as line, but the last and first nodes are
99     linked. In the previous example, 'a-b,␣b-c,␣c-d,␣d-a' links
100    would be created.
101
102    - pairs: The links would be created in pair. In the previous
103    example, 'a-b,␣c-d' links would be created.
104
105    - clique: The given nodes are linked to form a clique or
106    complete graph. In the previus example, 'a-b,␣a-c,␣a-d,␣b-c,␣b-d,
107 ␣␣␣c-d' would be created.
108
109 -->
110 <!ELEMENT link (node*)>
111 <!ATTLIST link
112           type (line|circle|pairs|clique) "line">
113 <!ELEMENT node EMPTY>
114 <!ATTLIST node
115           name CDATA #REQUIRED>
```

6.4 Game saving and loading

As this game matches can take very long, specially in big maps or in presence of multiple players, it is very nice to have a feature to store and load games.

To achieve this, the best system is to correctly separate the concerns of *state* and *behaviour* in the model, the former being intended to be stored persistently modelling an instantaneous image of the game world and the later concentrating on local and dynamic reactivity of the system. This allows us to easily use Python *object pickling* to automatically serialise the state and dump it into a file. For more information about object pickling visit:

`http://docs.python.org/library/pickle.html`

It is still unsure weather a second file providing metadata about the save game (possibly using XML) would actually be needed.

TO-DO note 13 (Juan Pedro) This last phrase probably should be changed into something concrete in the final version of the document.

6.5 Customisation

The application uses a customisation system that relies on the `ConfNode` class. This system provide a hierarchical multi-backend and observable configuration system. By hierarchical we mean that the configuration is composed of a tree of values, where a node in the configuration can have some other child values. By multi-backend we mean that the actual persistence of this configuration is hidden to most of part of the application and can be changed at runtime. This could allow to have a system that can either store the configuration using `gconf`, the Windows registry, `INI` files or `XML` files. Currently, a `XML` backend is implemented that is limited to `str`, `int`, `bool` and `float` values. By observable we ultimately mean that one can register slots (callbacks) into one node to be notified on updates, turning the configuration system into a *model* (using the MVC terminology) that aids in separating the concerns of the objects that modify the configuration. *controllers*, and the ones that whose behaviour depend on it, *views*.

The configuration nodes are named by a string, and the path to a node

can be encoded using the a dot `'.'` to relate a parent node to its child. The following sections describe the configuration values used by the application.

TO-DO note 14 (Juan Pedro) Should I make class diagrams of the `base` module that I imported from the other project so this paragraph is more clear? (It currently refers to classes, i.e. the ConfNode, that are not modelled anywhere!)

6.5.1 SFML configuration

These values are able to configure how the game shows on the screen. Probably most of them will be useless in the final release as the videogame console will use fixed parameters, but they are useful for debugging and can aid in a later PC port.

Option	Type	Meaning
`jagsat.sfml.width`	`int`	Screen width in pixels.
`jagsat.sfml.height`	`int`	Screen height in pixels.
`jagsat.sfml.bpp`	`int`	Colour depth in bits per pixel.
`jagsat.sfml.vsynch`	`bool`	Vertical synchronization activated.
`jagsat.sfml.fps`	`int`	Execution frames per second.
`jagsat.sfml.full`	`bool`	Use full-screen or windowed mode.

6.5.2 Player profiles

It is very useful to store game profiles. For example, a user Mike might usually play with his friends Tom and Anna some days, and other days with his father John and his mother Lisa. To avoid setting up the player position, colors, and the map that he likes to play in these contexts, he could use a *profile* called *Family* or *Friends* for either case.

To support this functionality with minimum code boilerplate our hierarchical customisation system can store this profiles in a node for each.

Option	Type	Meaning
`jagsat.profiles.`*`profile`*	`None`	A node containing all the options of a given profile *profile*.

Each profile contains all the options needed to set up a game. We will now shorten `jagsat.profiles.`*`profile`* as just *`profile`*.

Option	Type	Meaning
`profile``.player_1`	`None`	Options of the first player.
`profile``.player_2`	`None`	Options of the second player.
`...`	`...`	...
`profile``.player_6`	`None`	Options of the sixth player.
`profile``.map`	`str`	File name of the map.

For each player, we have the following set of options.

Option	Type	Meaning
`player_`*`n`*`.name`	`str`	Name of the player.
`player_`*`n`*`.color`	`int`	Colour of the player.
`player_`*`n`*`.position`	`int`	Position of the player around the table, where $position \in [1, 6]$.

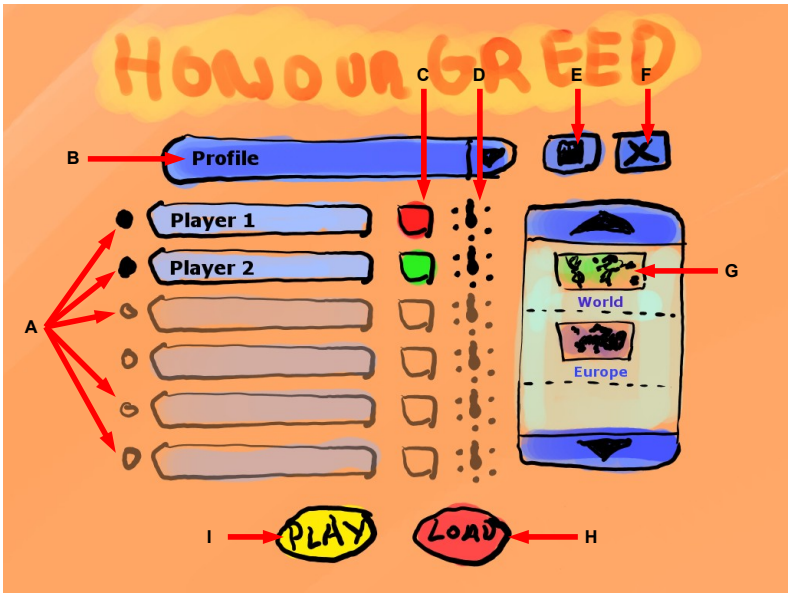
7 User interface

This section includes a series of mock-ups of the user interface. The different elements are tagged with letters and a table shows the behaviour associated with each element.

In some way, these mock-ups clearly reflect what is happening on the state machines, so it could ease the reader to go back and forward from/to the state machines while reading this section of the document to get a deeper understanding of the system.

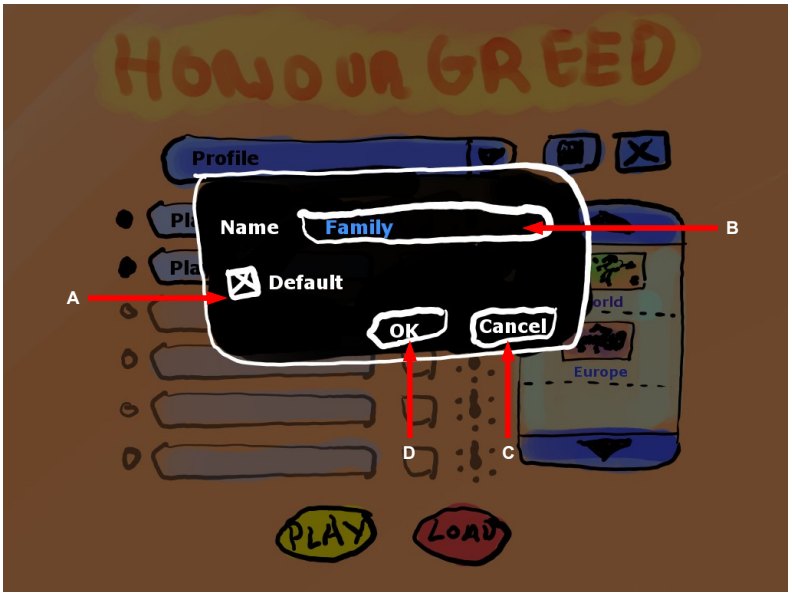
7.1 Main menu

7.1.1 Configuration



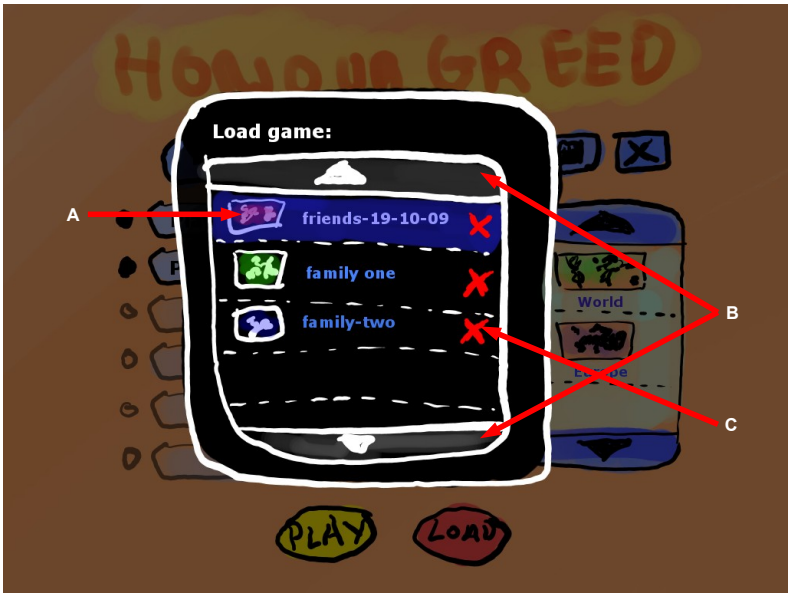
	Name	Action
A	Players configuration	By clicking the black left button the player will be activated. By clicking the player name button the keyboard will appear to change player name.
B	Profile menu	Displays profile menu with profiles list
C	Player colour configuration	By clicking the button player color changes
D	Player position configuration	Points the player position around the device
E	Save profile	Shows 7.1.2 menu
F	Delete profile	Deletes profile
G	Possible maps	Map selected will be remarked.
H	Load game	Shows 7.1.3 menu
I	Play game	Starts game if the actual config is valid.

7.1.2 Save profile



	Name	Action
A	Set profile as default	This profile will be shown as marked in 7.1.1B
B	Profile name.	
C	Cancel button:	Back to 7.1.1 without changes
D	Ok button	Back to 7.1.1 with profile changes

7.1.3 Load game



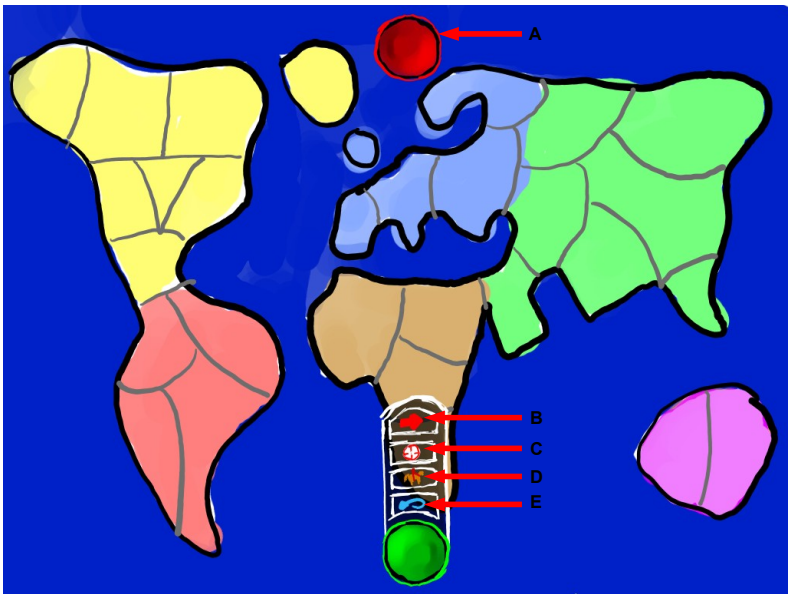
	Name	Action
A	Map preview	Little picture of the kind of map played in this game.
B	Up/Down buttons	
C	Delete saved game.	

7.1.4 Confirmation

	Name	Action
A	Confirm	
B	Cancel	

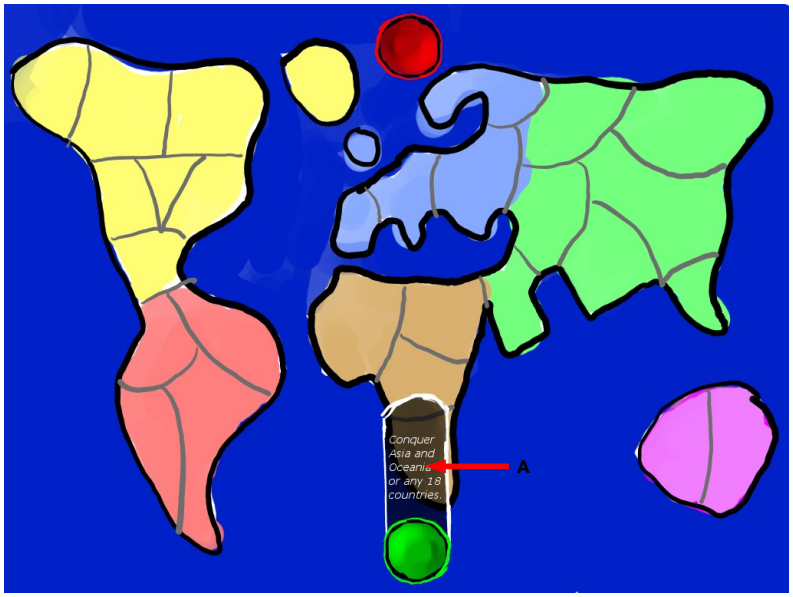
7.2 In game player menu

7.2.1 Overview



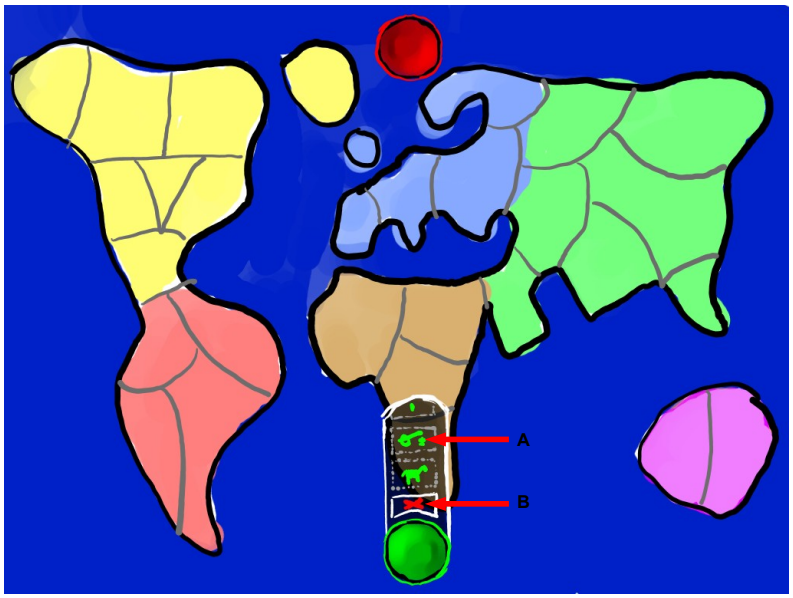
	Name	Action
A	Player menu button	Shows the menu with options [B,E]
B	”Next” button	Perform the action “next” if avail[able
C	”Objective” button	Shows objective 7.2.2
D	”Cards” button	Shows cards 7.2.3
E	”Undo” button	Undo the previous action.

7.2.2 Objective



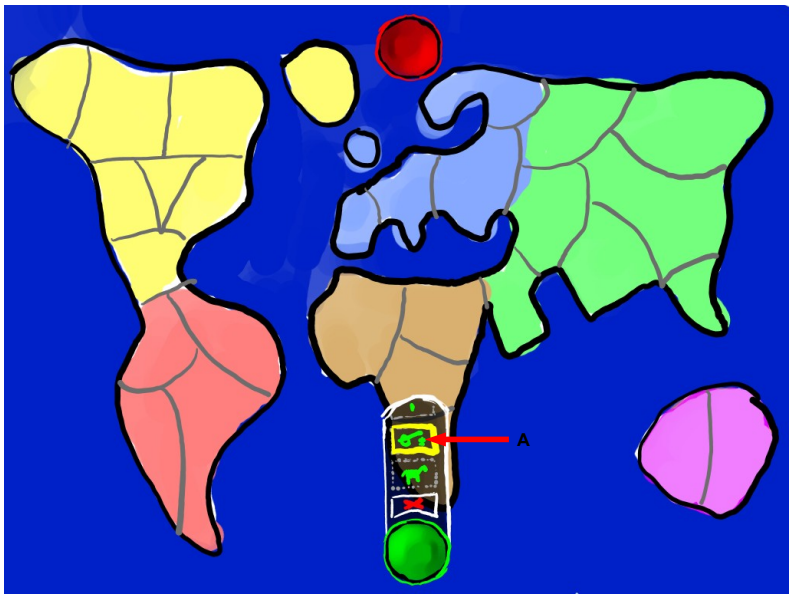
	Name	Action
A	Objective text	Hides the objective text

7.2.3 Cards



	Name	Action
A	Cards	Select the cards that the player owns.
B	”Exit” button	Back to 7.2.1

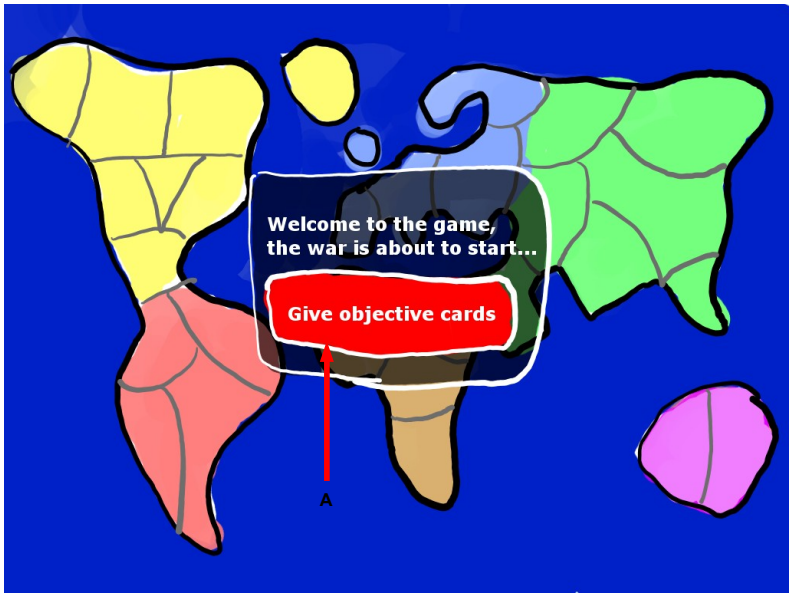
7.2.4 Cards selection



	Name	Action
A	Card	Unselected card.

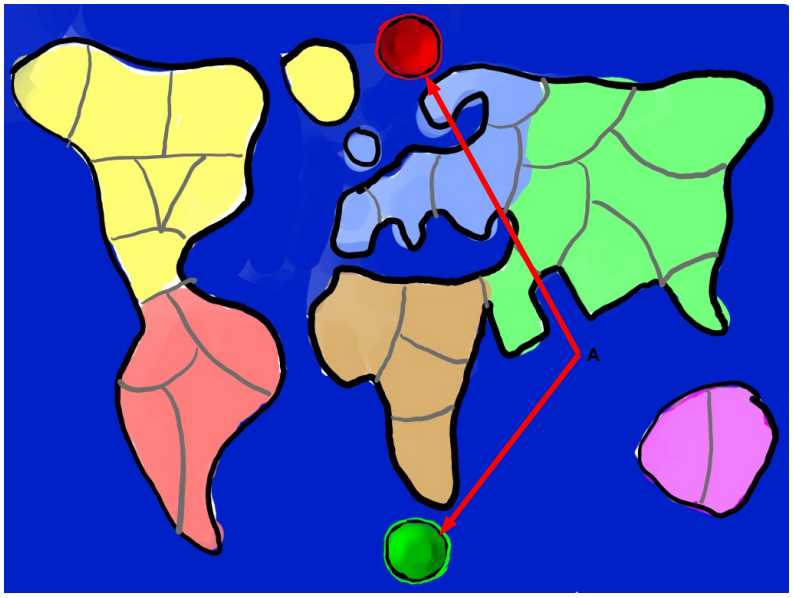
7.3 Init phase

7.3.1 Objectives deal



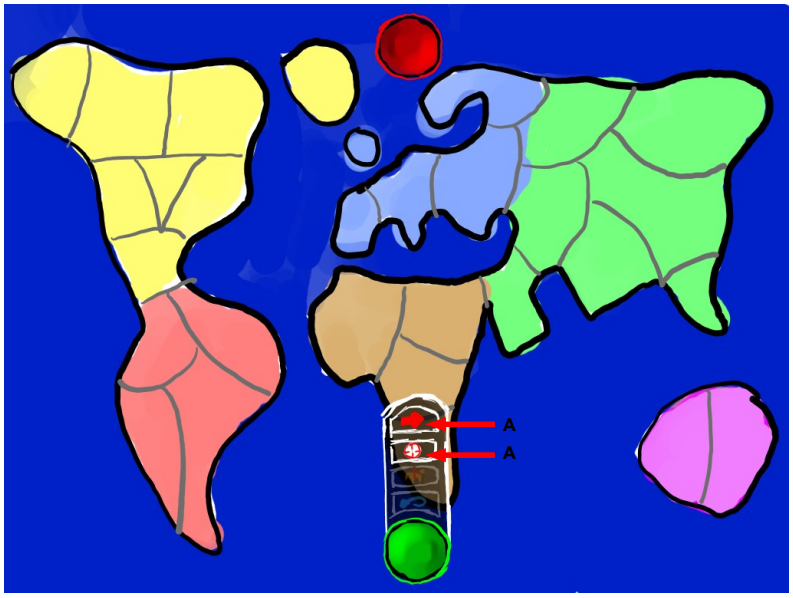
	Name	Action
A	Give objective	Deals an objective to each player.

7.3.2 Player positions



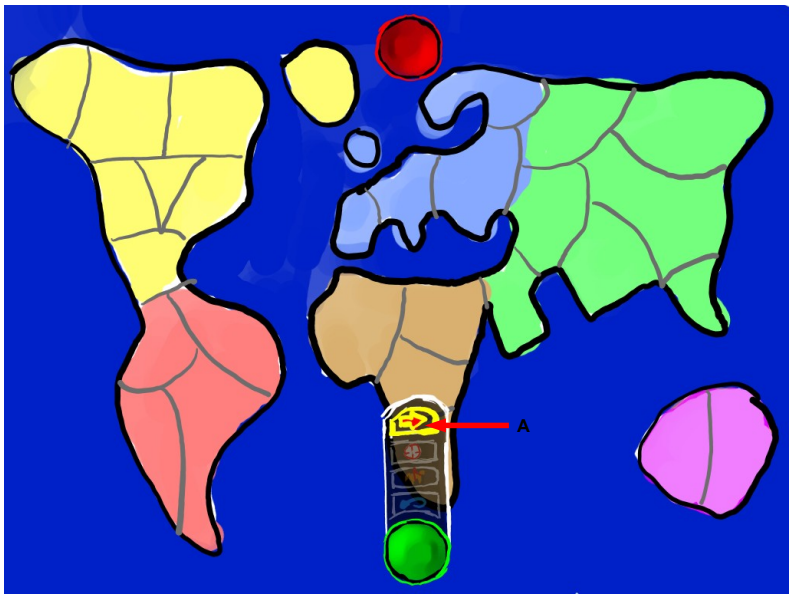
	Name	Action
A	Player menu	Display player menu 7.2

7.3.3 View objective active



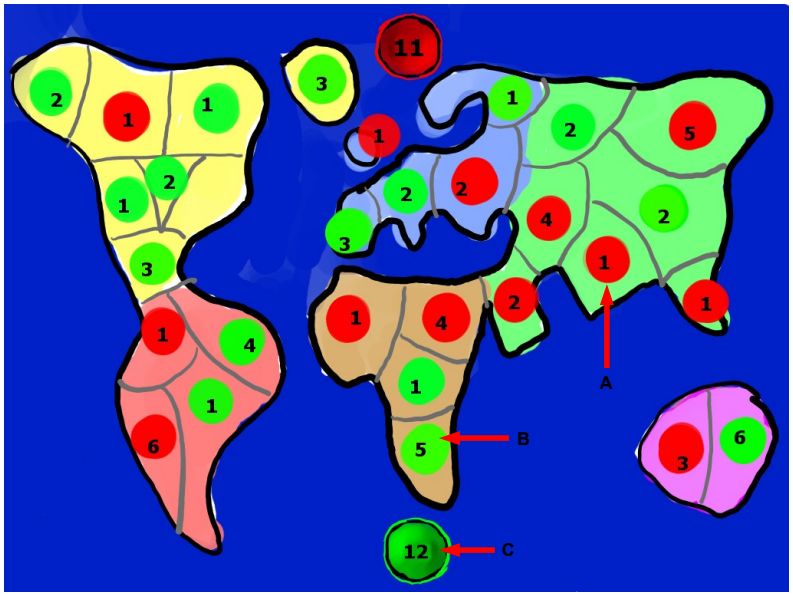
	Name	Action
A	Next	Goes to 7.3.4
B	Show objective	Goes to 7.2.2

7.3.4 Next button active



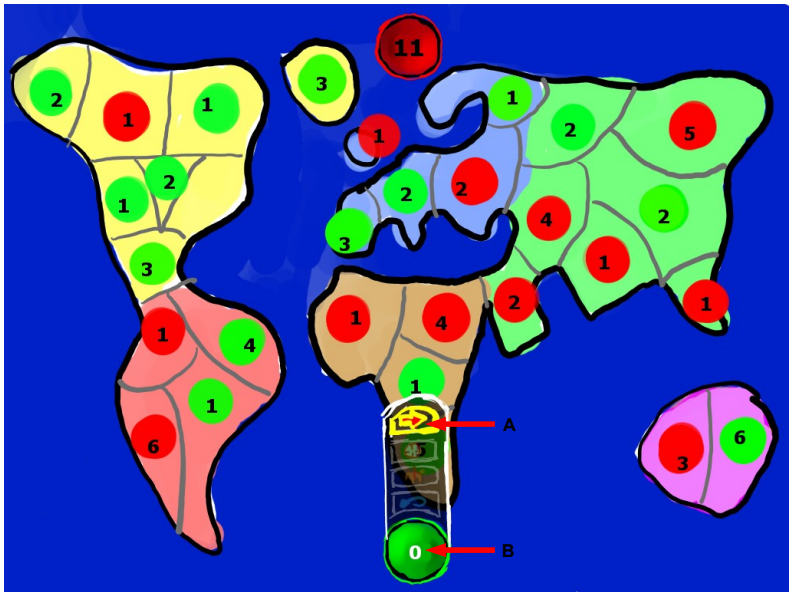
	Name	Action
A	Next button selected	Goes to 7.3.5

7.3.5 Troops placement



	Name	Action
A	Red region troops	Increases number of red troops.
B	Green region troops	Increases number of green troops
C	Troops left	

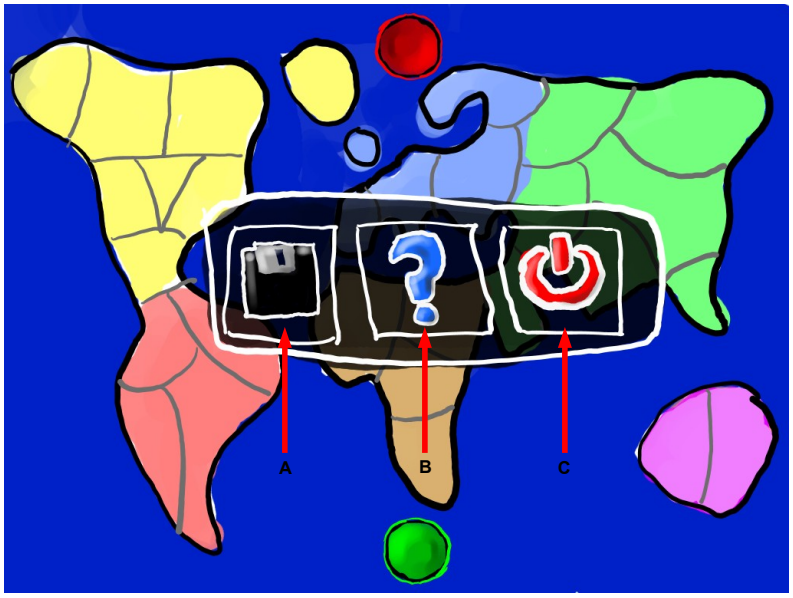
7.3.6 Finish troop placement - Next button active



	Name	Action
A	Next button	Finish init phase.
B	Green region troops	Activates next button when reaches 0.

7.4 In-game menu

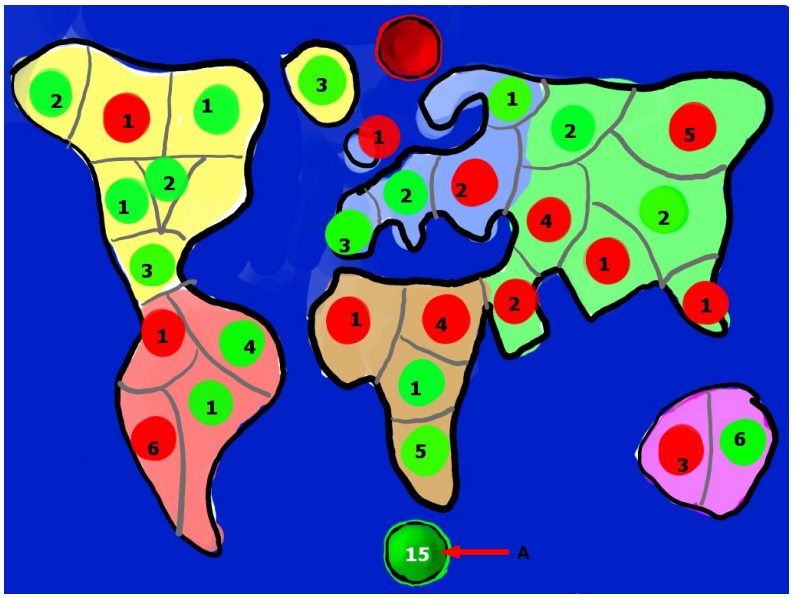
7.4.1 Showing in-game menu



	Name	Action
A	Save	Saves the game.
B	Help	Shows the help menu.
C	Power	Finish the game.

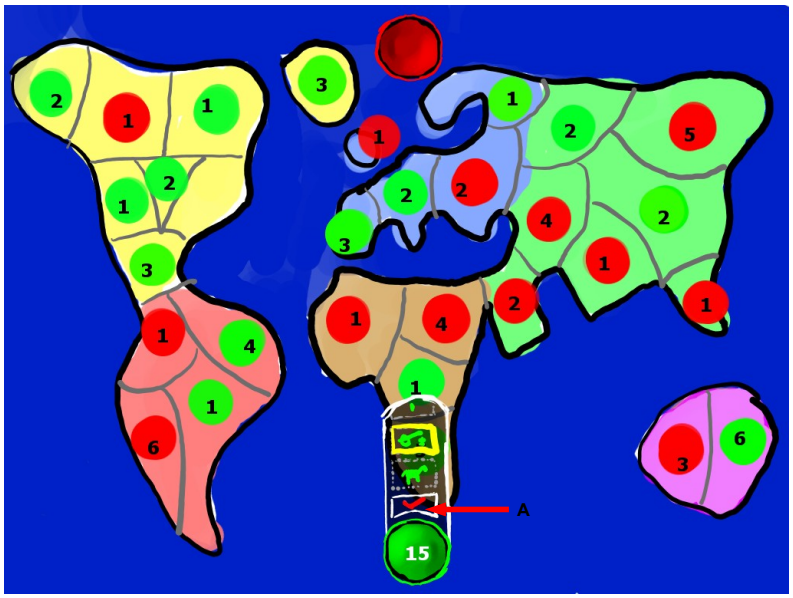
7.5 Game Round

7.5.1 Reinforcements



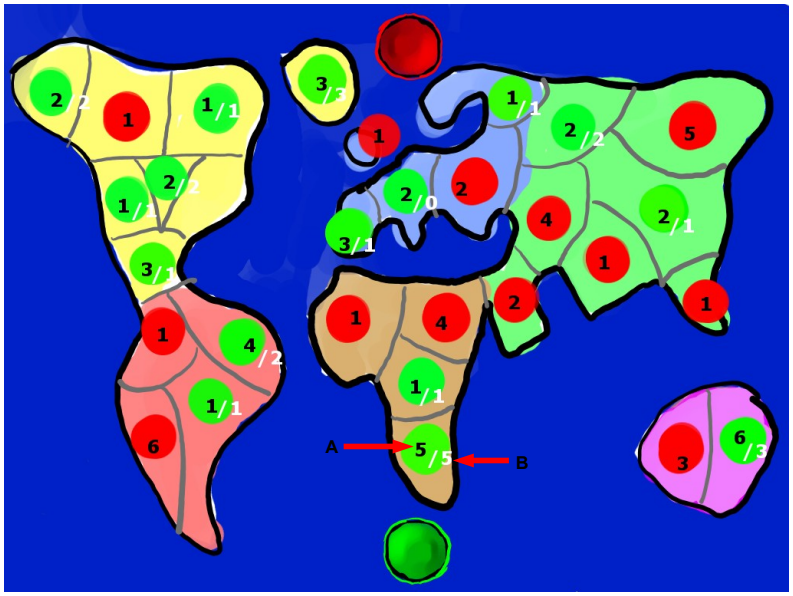
	Name	Action
A	Troops left	

7.5.2 Cards use



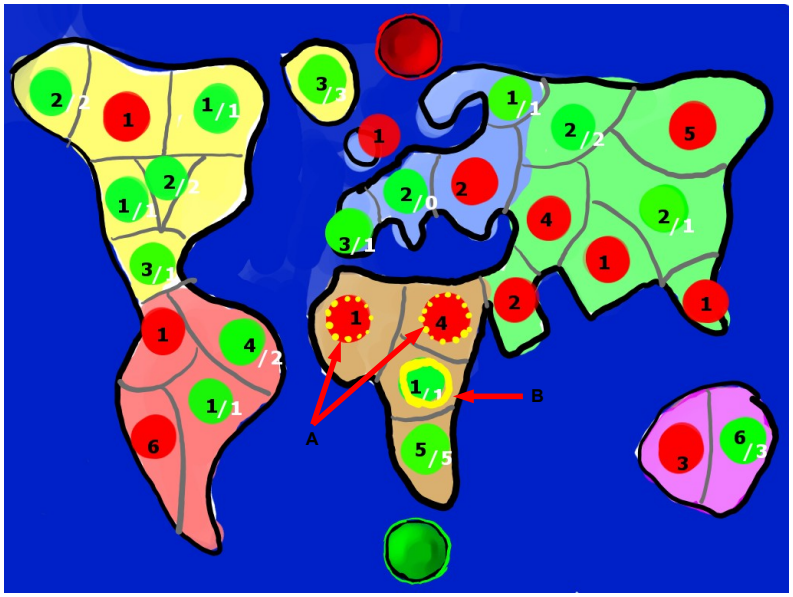
	Name	Action
A	Done button	If the card selected make a valid combination, the result of troops is add to the number available.

7.5.3 Attack phase



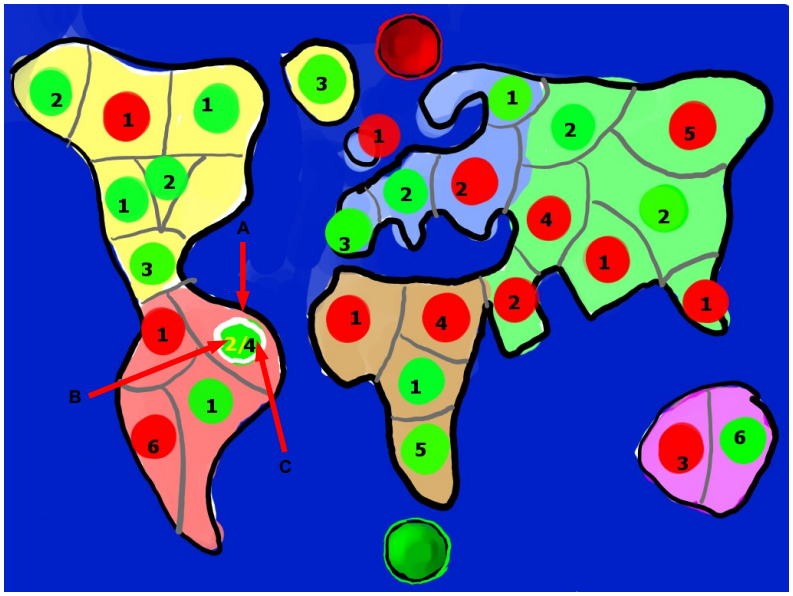
	Name	Action
A	Troops in region	
B	Troops available.	Troops available to use in an attack.

7.5.4 Attack phase - Region selected



	Name	Action
A	Defenders	Regions that are able to be attacked
B	Attacker	Region that is going to attack. Makes "Defenders" more visible when clicked.

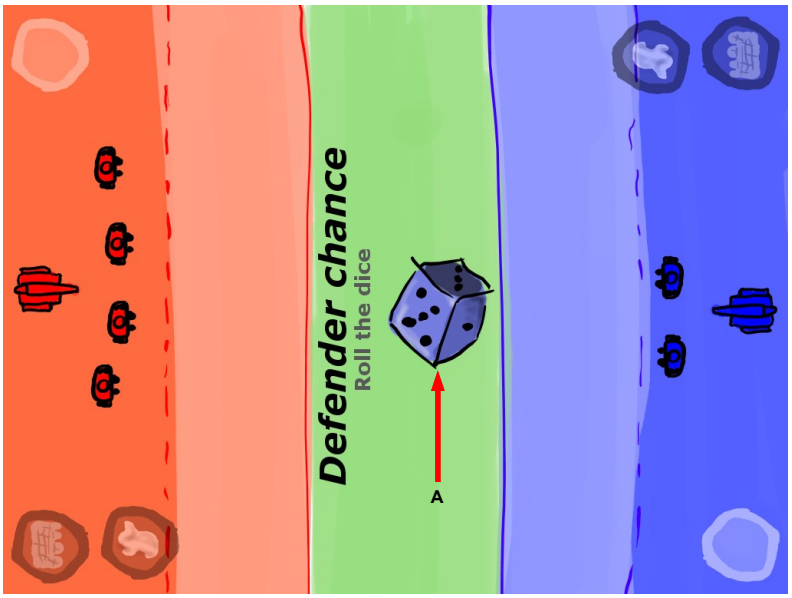
7.5.5 Movement phase



	Name	Action
A	Selected region	
B	Moveable units	
C	Total units	

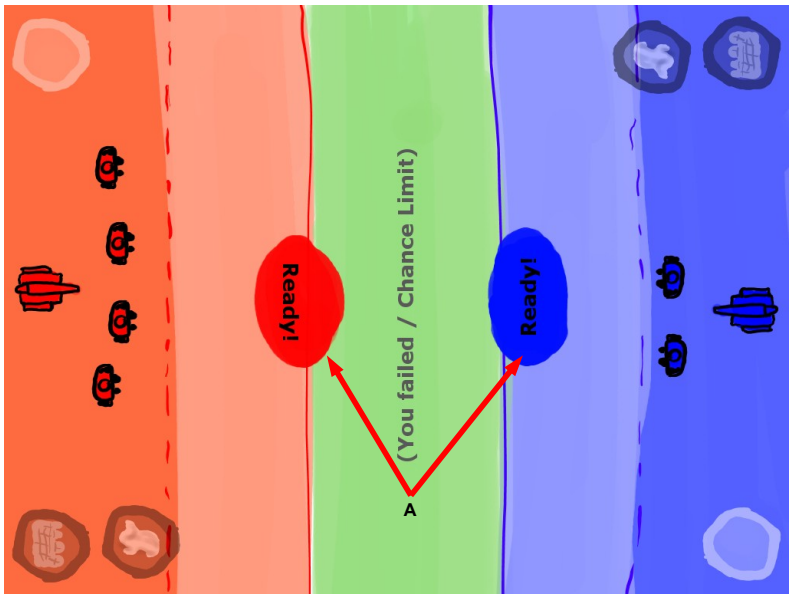
7.6 Battle

7.6.1 Defender reinforcements



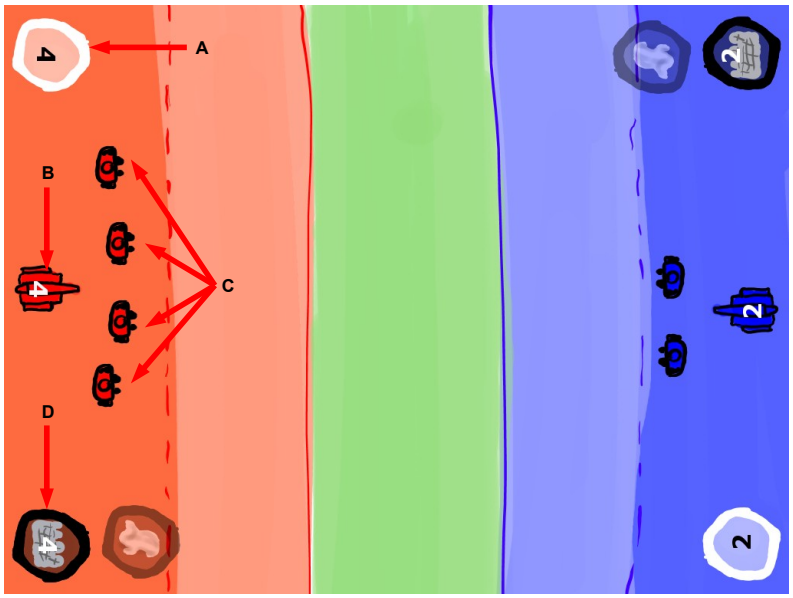
	Name	Action
A	Dice	Roll when clicked.

7.6.2 Defender reinforcement ends



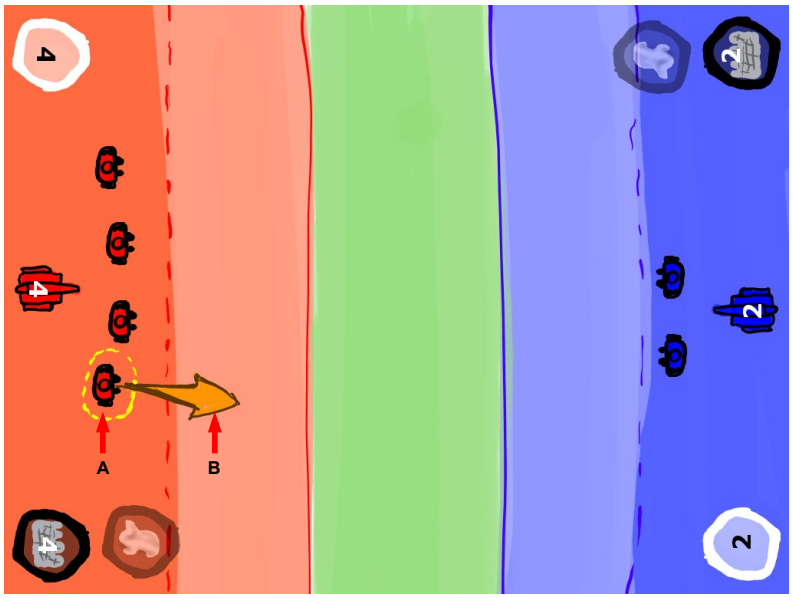
	Name	Action
A	Ready button	Starts the battle after both players have clicked the button.

7.6.3 Battle starts



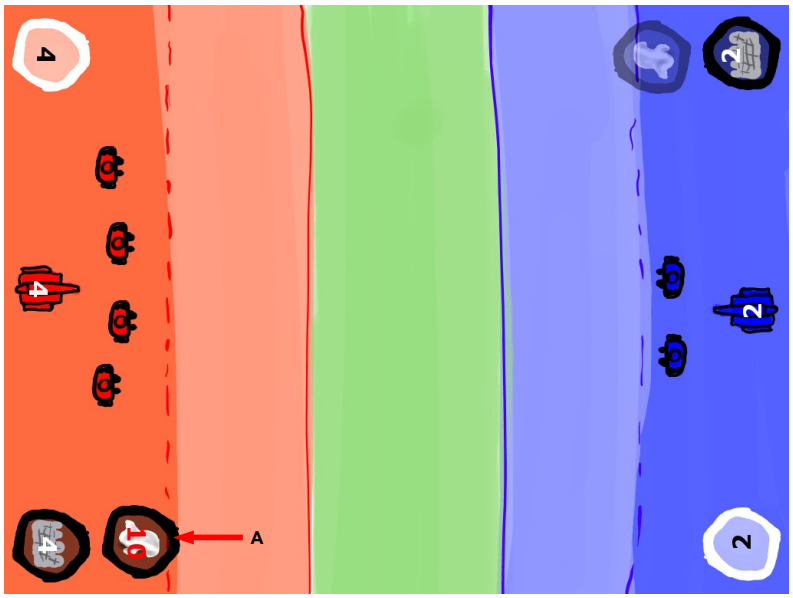
	Name	Action
A		
B	Cannonballs available	Shoots cannon.
C	Troops available	Goes to 7.6.4
D	Walls available	Goes to 7.6.6

7.6.4 Soldier movement



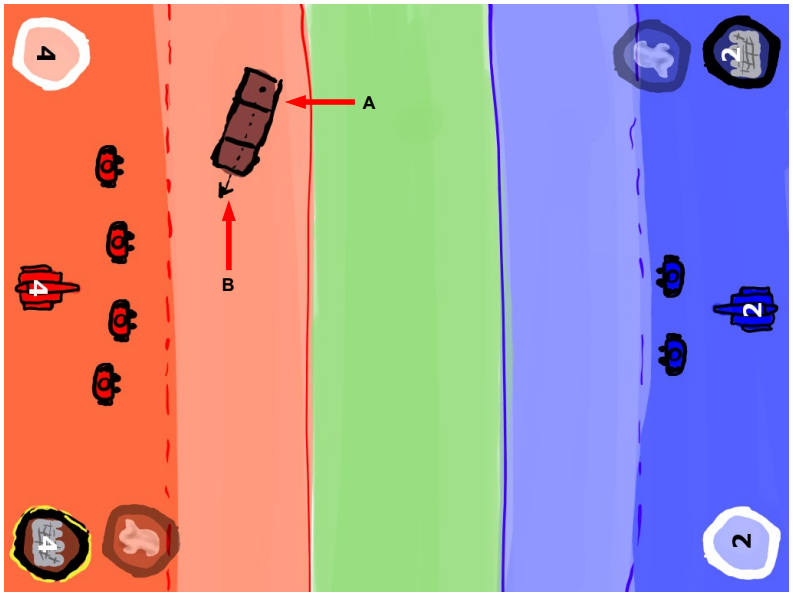
	Name	Action
A	Troop selected	Shows direction arrow.
B	Direction arrow	Moves the troop in the selected direction.

7.6.5 Retreat activates



	Name	Action
A	Retreat countdown	Retreats the attacker if clicked when available.

7.6.6 Wall placement



	Name	Action
A	Wall	
B	Wall direction	Click and drag to place the wall.

TO-DO note 15 (Juan Pedro) I see that there has been some misunderstandings between me (who draw the mocks) and Alberto (who modelled the behaviour) and that the intended behaviour of some parts of the mock-ups where not obvious. I will try to fix it some other day :p

Appendices