

Specification for Marvelous Mashup

Kamila Datbayev, Merten Dieckmann, Emilija Kastratovic,
Marlene Mika, Benjamin Raible, Markus Thielker

Contents

1	Introduction	4
1.1	Introduction	4
1.2	Motivation	4
1.3	Vision	4
1.4	Project context	4
2	Terms	4
2.1	Actors	5
2.2	Technical knowledge	6
3	Requirements	13
3.1	Functional requirements	13
3.1.1	Server	13
3.1.2	Client	18
3.1.3	AI-Client	21
3.1.4	Editor	22
3.2	Non-functional requirements	22
4	Usecases	24
4.1	Use Case Diagramm	24
4.2	Component behavior	24
4.2.1	Client	24
4.2.2	Server	25
4.2.3	Editor	25
4.3	Gameplay	26
4.3.1	Client registration	26
4.3.2	Game preparation	26
4.3.3	Assemble superhero squad	27
4.3.4	Game round	27
4.3.5	Pause handling	28
4.3.6	Check win condition	29
4.3.7	Overtime mechanism	30
4.4	AI-Client	30
4.4.1	Startup	30
4.5	Changing configurations	30
5	Models and diagrams	31
5.1	Domain model	31
5.2	Use case diagram	32
6	Mock-Ups	34
6.1	Dialog	34
6.1.1	UI Element Overview	34
6.1.2	Client	34
6.1.3	Dialog Structure	34

6.1.4	Mockups	35
6.1.5	Editor	41
6.1.6	Dialog Structure	41
6.1.7	Mockups	41
6.1.8	Server	42
6.2	Gameboard	42
7	Acceptance criteria	45

1 Introduction

1.1 Introduction

In this project, we work on developing a multiplayer round-based tactic-game called “Marvelous Mashup”. That includes the organization and implementation of the functional requirements. The participants shall gain theoretical and practical experience within the parallel lecture and the realization of this game. Finally, the game is presented to its customer and should run without any errors or bugs. Especially the proper fulfillment of the requirements and wishes provided by the stakeholder is targeted by the team.

1.2 Motivation

The project idea was planned and created by a computer science course to further implement an actual product with predefined functions and expectations. In the first half, the team has to design the realization facing the requirements provided, while in the second part, the actual development occurs. Thus the approval of the provided service for the finished product is given by the client. Because of the immense success of the Marvel Cinematic Universe, this game has great potential to appeal to a broad audience.

1.3 Vision

The completed project will be a round-based strategy game taking place in the setting of the Marvel Universe. It will gamify the hunt for a full collection of the Infinity Stones, a set of valuable items providing special powers to its owner. Each of the two players needed to play a match will choose their team compiled from a pool of superheroes to collect all infinity stones and gain victory. Strategic thinking is demanded from the players by the complex movement and action system. Each superhero has a predefined movement and action points per round and a total amount of health points. These character-specific limitations request tactical choices when planning the next moves to attack the enemy’s figures. Action-points are spent by choosing to attack from range, in close combat, or using one of the special powers given by one of the infinity stones in the character’s inventory, just as passing these to an ally. The game contains six colored stones, each capable of unique special action. The map the competition is taking place on is based on a simple plain setting. This allows users to keep a clear overview of the at any time fully revealed gameplay.

1.4 Project context

The realization of the project takes place parallel to the course “Software Engineering”. Involved stakeholders are supervisor Florian Ege and assigned tutor Bastian Wankmüller.

A young and highly motivated team fits perfectly into the requirements needed to proceed the just described concept into a powerful game based on modern development methods from the structural and technical perspective. The contemporary setting provides the perfect entry point to a long-term expansion possibility, with more game modes, pickable characters, and more content such as possible franchise collaborations to come.

2 Terms

It is important to define all the terms whose context-dependent interpretation is either not identical for all parties involved in project development or familiar at all.

2.1 Actors

Term	User
DESCRIPTION:	Every person interacting with the application is called a user.
IS-A:	Human
CAN-BE:	Player, Spectator
EXAMPLE:	-

Term	Player
DESCRIPTION:	Every user actively participating in a match is called a player.
IS-A:	Human, User, AI-Client
CAN-BE:	-
EXAMPLE:	-

Term	Spectator
DESCRIPTION:	Every user, passively viewing a match is called a spectator.
IS-A:	Human, User
CAN-BE:	-
EXAMPLE:	-

Term	AI-Client
DESCRIPTION:	An autonomously playing client, meaning not controlled by a human, entering the role of a player (max. 1 per match).
IS-A:	-
CAN-BE:	Player
EXAMPLE:	-

Term	Client
DESCRIPTION:	The client is the part of the system, used by users to interact and communicate with the servers to play the game. It also shows them the current state of the game on a graphical interface.
IS-A:	-
CAN-BE:	Player, Spectator
EXAMPLE:	-

Term	Server
DESCRIPTION:	The server is the part of the system running the game logic, processing connections and receiving messages from the clients.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Standardization-Committee
DESCRIPTION:	The standardization-committee is an instance, deciding about communicational conditions, required for implementation.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

2.2 Technical knowledge

Term	Match
DESCRIPTION:	A match is the active application of the game logic with two players.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Win conditions
DESCRIPTION:	<p>The win conditions are considered in the following order:</p> <ul style="list-style-type: none">• Primary win condition: Meaning that one character has collected all infinity stones. If that's the case, the server ends the match and informs all clients accordingly.• Secondary win condition: The group that possessed the higher amount of Infinity Stones at a certain point in time wins.• Tertiary win condition: The group that knocked out the higher amount of opponent characters wins.• Quaternary win condition: The group that caused the higher amount of Health Point (HP) damage to the opponent team in total wins.• Quinary win condition: If all of the above ends up as a tie the winner is drawn randomly.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Character
DESCRIPTION:	<p>A character is an object with predefined values set in a character configuration file. These are:</p> <ul style="list-style-type: none"> • Name: Each character has its name. • Avatar: Each character is represented by an avatar to distinguish between the different participating characters. • Movement Points (MP) and Action Points (AP): Each character receives a certain number of points per turn, which can be used for movement steps and actions. • Health Points (HP): HPs represent the health status of the character. If the HPs drop to 0, the character is knocked out and can't make any more moves until he is revived. If a character has Infinity Stones in his inventory when being knocked out, these stones are placed in random order on random free neighboring fields of the character. • Inventory: A character can carry Infinity Stones in their inventory.
IS-A:	-
CAN-BE:	-
EXAMPLE:	<p>Rocket Raccoon: HP: 100 MP: 2 AP: 2 Close combat: Damage 10 Range combat: Damage 30, Range 5</p>

Term	Character pool
DESCRIPTION:	The character pool describes the total of all loaded characters.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Goose
DESCRIPTION:	Goose is a pre-made non-playable character, controlled by the computer, exclusively responsible for distributing the infinity stones on the board. To do this, she teleports to a random square in each of the first six rounds and places a random stone.
IS-A:	non-playable character
CAN-BE:	-
EXAMPLE:	-

Term	Thanos
DESCRIPTION:	Thanos is a pre-made non-playable character controlled by the computer, appearing at the beginning of a game and is responsible for ending an overlong game.
IS-A:	non-playable character
CAN-BE:	-
EXAMPLE:	-

Term	Stan Lee
DESCRIPTION:	Stan Lee is a pre-made non-playable character controlled by the computer, not relevant to the gameplay, but he is bringing not further specified Easter-eggs to the application.
IS-A:	non-playable character
CAN-BE:	-
EXAMPLE:	-

Term	Line of sight
DESCRIPTION:	There is a line of sight between two fields if there is no blocking field between fields A and B, considering the path with the smallest distance.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Distance
DESCRIPTION:	Minimum number of consecutive steps to neighboring fields (in all eight directions) to get from A to B.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Intro spot
DESCRIPTION:	The intro is an animation explaining what had previously happened. Thanos had gathered all the Infinity Stones when Goose appeared and swallowed them.
IS-A:	pre-match Round
CAN-BE:	-
EXAMPLE:	-

Term	Assemble superhero squad
DESCRIPTION:	The process of the election phase from a player's point of view is like this: <ol style="list-style-type: none"> 1. The server takes 12 random characters from the character pool and displays them to the player 2. The player chooses 6 characters for his team.
IS-A:	pre-match Round
CAN-BE:	-
EXAMPLE:	-

Term	Character placement
DESCRIPTION:	All characters are placed randomly on free fields on the game board.
IS-A:	pre-match Round
CAN-BE:	-
EXAMPLE:	-

Term	Goose Rounds
DESCRIPTION:	At the beginning of rounds 1 to 6 (i.e. before the characters make their moves), Goose appears on a random free field, throws up a random Infinity Stone, still in her stomach on that field, and disappears again.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Stan Lee Round
DESCRIPTION:	At the end of the first non-Goose round, Stan Lee appears on a free neighboring field of the character with the lowest HPs. All characters in line of sight of Stan Lee get back to full HP, knocked-out characters get revived. Stan Lee yells "Excelsior!" and disappears.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Round
DESCRIPTION:	A round describes every character, not knocked out, making his move. The order, the characters a making their moves in, is created randomly.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Move
DESCRIPTION:	A move consists of multiple moving phases, moving- and action-points can be spent in.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Moving phase
DESCRIPTION:	In every moving phase the player can decide to move his character or execute an action.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Game board
DESCRIPTION:	The game board is a rectangular Cartesian grid that consists of x*y chessboard-like squares.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Field
DESCRIPTION:	<p>Each field on a board can have 4 states which are defined as follows:</p> <ul style="list-style-type: none"> • grass: free field, characters can stand on it and move over it. • rock: blocked field, characters can't stand on it or move over it. A Rock can be destroyed by attack them using AP. A Rock got 100HP. • grass with character: characters can stand on it and move over it. When field gets accessed the moving character and the character standing on the field switch positions. • grass with infinity stone: characters can stand on it and move over it. When field gets accessed, the moving character picks up the infinity stone automatically.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Action
DESCRIPTION:	<p>The following moves can be performed by spending APs:</p> <ul style="list-style-type: none"> • Close combat attack: The attack of an enemy character, if the attacked character is inside of a distance of 1 field. The enemy character has the close combat damage of the attacker subtracted from its Health Points (HP). • Ranged combat attack: The attack of an enemy character, if the attacked character is inside of a range of 2 and the defined maximal range of fields and in line of sight of the attacker. The opponent has the damage subtracted from its Health Points (HP). • Usage of Infinity Stones: The character uses the special ability from an Infinity Stone in his inventory. • Transfer of Infinity Stones: The character passes an Infinity Stone to another character from his team, standing on a neighboring field.
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

Term	Infinity Stones
DESCRIPTION:	<p>Infinity stones are needed to reach the main win condition and furthermore apply the usage of special actions to the character, carrying it in its inventory. The different stones and actions are:</p> <ul style="list-style-type: none"> • Space Stone (blue): The carrier can teleport to any free field. • Mind Stone (yellow): The carrier can focus their mental energy into a beam and do a ranged combat attack on any target in line of sight. • Reality Stone (red): The carrier can make a rock disappear on an adjacent field or make a rock appear on an free adjacent field (with 100 HP). • Power Stone (purple): The carrier can perform a close combat attack that deals twice as much damage as its normal attack, but reduces the carrier's HP by 10% of its maximum HP, with a lower limit of one HP. • Time Stone (green): The carrier can "rewind time" using one AP and by that reset its MPs and APs to the initial value from the beginning of the round. E.g., a character with 2 MP and 2 AP can move twice, hit someone with an action, and use the Time Stone with their last AP, and then have 2 MP and 2 AP again with which they can continue their turn. • Soul Stone (orange): The carrier can revive a knocked-out character on an adjacent field using one AP. The revived characters' HPs are set to the full value specified in the character configuration. <p>After using the special action of the stone, it goes into cooldown for the amount of time individually specified for each stone. The cooldown remains active when the stone gets passed to another character.</p>
IS-A:	-
CAN-BE:	-
EXAMPLE:	-

3 Requirements

3.1 Functional requirements

This section contains the requirements imposed on the system and gives a detailed description of the systems' functionality.

3.1.1 Server

ID	FR001
TITLE:	Supported operating system
DESCRIPTION:	The server must be startable in the Linux command line in form of a Docker-container.
EXPLANATION:	A Docker-container helps making the application independent from the specific operating system.
REFERENCES:	-
PRIORITY:	++

ID	FR002
TITLE:	Match configuration
DESCRIPTION:	The server loads a match configuration at the start.
EXPLANATION:	The match configuration contains parameters for each match, such as periods allowed per action and the number of rounds until the overtime condition for a match is met.
REFERENCES:	FR042, FR043
PRIORITY:	++

ID	FR003
TITLE:	Scenario configuration
DESCRIPTION:	The server loads a scenario configuration at the start.
EXPLANATION:	The scenario configuration defines, what the game board looks like.
REFERENCES:	FR042, FR043
PRIORITY:	++

ID	FR004
TITLE:	Character configuration
DESCRIPTION:	The server loads a character configuration at the start.
EXPLANATION:	The character configuration contains all the selectable characters and their values.
REFERENCES:	FR042, FR043
PRIORITY:	++

ID	FR005
TITLE:	Player connection
DESCRIPTION:	The server allows exactly two players to connect to a match. Remark: The players can be either users or AI-Clients.
EXPLANATION:	Marvelous Mashup is a two player game, so exactly two clients are needed to play a game.
REFERENCES:	FR014, FR020, FR037
PRIORITY:	++

ID	FR006
TITLE:	Spectator connection
DESCRIPTION:	Users can connect to the server as spectators. That's still possible even if the game is already running. They get sent the current state of the game as well as future updates.
EXPLANATION:	Every user should be able to spectate any match.
REFERENCES:	-
PRIORITY:	++

ID	FR007
TITLE:	Web Socket
DESCRIPTION:	The port for the web socket connection should be 1218.
EXPLANATION:	This wish was provided by the customer.
REFERENCES:	FR005, FR006, FR020, FR037
PRIORITY:	-

ID	FR008
TITLE:	Log file
DESCRIPTION:	The server writes all information about a match in a log file. Remark: The format is to be decided by the standardization-committee.
EXPLANATION:	This enables users to watch the replay of a match.
REFERENCES:	FR031
PRIORITY:	-

ID	FR009
TITLE:	Information publishing
DESCRIPTION:	The server notifies all connected clients about the players' actions, the thereby triggered events, and the resulting state of the match.
EXPLANATION:	Every user should get updated about what happens in the match.
REFERENCES:	FR021, FR024, FR028
PRIORITY:	++

ID	FR010
TITLE:	Timeout
DESCRIPTION:	When a client is required to send a message to the server in a certain amount of time, but the server does not receive this message, the connection to the client should be terminated. If the client is one of the players, they get automatically disqualified.
EXPLANATION:	User who lost connection should be detected and removed.
REFERENCES:	FR012
PRIORITY:	++

ID	FR011
TITLE:	Delayed messages
DESCRIPTION:	When a message from an earlier round reaches the server, it shouldn't perceive it as a violation of the protocol but ignore it.
EXPLANATION:	-
REFERENCES:	-
PRIORITY:	++

ID	FR012
TITLE:	Reconnect
DESCRIPTION:	If a client loses connection to the server, he has a certain amount of time to reconnect. If he does, he gets receives the current state of the match.
EXPLANATION:	To ensure players a good game experience, momentarily disconnects shouldn't end the match.
REFERENCES:	FR010, FR033
PRIORITY:	++

ID	FR013
TITLE:	Protocol violation
DESCRIPTION:	If a client violates the communication protocol or try's to make an invalid move, the server shall be notified about the violation to terminate the users connection. If the client was a player his opponent wins by default.
EXPLANATION:	The server can only tolerate valid moves and actions.
REFERENCES:	FR005, FR006, FR011, FR038
PRIORITY:	++

ID	FR014
TITLE:	Match start
DESCRIPTION:	When two players are connected to the server, the match gets started.
EXPLANATION:	As soon as there are enough players to play (2 players) the game should start.
REFERENCES:	FR005
PRIORITY:	++

ID	FR015
TITLE:	Squad assembly
DESCRIPTION:	In front of every match, both players get a list of twelve superheroes they can assemble their squad out of six heroes from. The given selection was randomly created by the server, based on the character pool inside the character configuration file.
EXPLANATION:	This selection phase removes the absolute randomness in the creation and provides the players with the ability to create strategies using their favorite superheroes.
REFERENCES:	-
PRIORITY:	++

ID	FR016
TITLE:	Goose Round
DESCRIPTION:	Each of the first six rounds of the game phase, starting right after the superhero assembly, begin with an animation of goose teleporting to a random position on the game board and throwing up a infinity stone.
EXPLANATION:	It is supposed to act as an introduction to the small story background of the game and for making the infinity stone placement more interesting.
REFERENCES:	-
PRIORITY:	++

ID	FR017
TITLE:	Win condition
DESCRIPTION:	The server tests after each move if the win condition is fulfilled.
EXPLANATION:	The game should end immediately, once the win condition is fulfilled.
REFERENCES:	FR029
PRIORITY:	++

ID	FR018
TITLE:	Overtime mechanism: Thanos
DESCRIPTION:	<p>Thanos appears and targets the next field where are Infinity Stones. In that round he has as many MPs as the character that has the most MPs.</p> <ul style="list-style-type: none"> • Thanos ignores obstacles. He can move through rocks which then crumble and turn to grass fields • If he moves to a field with an Infinity Stone that is on the floor, he adds it to his inventory and ends his turn • If he moves to a field with a character that has Infinity Stones, said character is moved to Thanos' previous position, is knocked-out, and all Stones are placed into Thanos' inventory and Thanos ends his turn • Thanos is immune to all attacks and can't be knocked-out • Characters can't move to a field Thanos is on <p>This is repeated until Thanos possesses all Infinity Stones. Meanwhile, his MPs are increased by one with every turn.</p>
EXPLANATION:	Accelerates game to come to an end after set #MAX_ROUND.
REFERENCES:	-
PRIORITY:	++

ID	FR019
TITLE:	Rule compliant game sequence
DESCRIPTION:	<ol style="list-style-type: none"> 1. Intro spot 2. Assemble superhero group 3. Character placement 4. Goose Rounds (Rounds 1-6) 5. Stan Lee Round 6. Rounds
EXPLANATION:	This specifies how a match is supposed to work.
REFERENCES:	-
PRIORITY:	++

3.1.2 Client

ID	FR020
TITLE:	Data delivery
DESCRIPTION:	On connection build up to the server, the client sends the server the currently set username, as well as the information if it is an AI-Client or controlled by a human.
EXPLANATION:	Spectators can distinguish between the participating players.
REFERENCES:	FR005, FR007
PRIORITY:	++

ID	FR021
TITLE:	Graphical user interface
DESCRIPTION:	The client visualizes the gameplay. The board, the players including their representing colors are shown on the interface.
EXPLANATION:	The spectators can see the different characters and identify to which player they belong.
REFERENCES:	FR009, FR023, FR024, FR028, FR022
PRIORITY:	++

ID	FR022
TITLE:	Stan Lee Easter Egg
DESCRIPTION:	There's a Stan Lee Easter-egg hidden on the graphical interface.
EXPLANATION:	This is added due to the set requirements by the customer.
REFERENCES:	FR021
PRIORITY:	-

ID	FR023
TITLE:	Gameplay information
DESCRIPTION:	Values and conditions of the characters, such as health bar, icons for the infinity stones in the inventory, a panel for MPs, APs, and the damage values, are shown.
EXPLANATION:	These values are necessary to understand the gameplay.
REFERENCES:	FR021, FR024
PRIORITY:	++

ID	FR024
TITLE:	Attack visualization
DESCRIPTION:	Close and range combat attacks are visualized by changes made to the health bar of the character that has been attacked.
EXPLANATION:	The spectators can see whos attacked and the resulting damage dealt to the victim.
REFERENCES:	FR009, FR021, FR023, FR028
PRIORITY:	++

ID	FR025
TITLE:	Possible actions
DESCRIPTION:	The possible actions get displayed in the respective phases of a match.
EXPLANATION:	This prevents the execution of illegal actions.
REFERENCES:	FR013
PRIORITY:	++

ID	FR026
TITLE:	Game tips
DESCRIPTION:	The user has the possibility to request tips from the game.
EXPLANATION:	For players learning the game, examples for recommended moves might be helpful.
REFERENCES:	FR040
PRIORITY:	++

ID	FR027
TITLE:	Settings Hotkeys
DESCRIPTION:	The user has the possibility to customize the controls by setting hotkeys.
EXPLANATION:	This feature provides a more comfortable user experience.
REFERENCES:	-
PRIORITY:	-

ID	FR028
TITLE:	Action animation
DESCRIPTION:	The client supports animations for executed actions. Remark: The duration of the animation must be adjusted to the length of a turn.
EXPLANATION:	The users should have a visual representation of what is happening in the match.
REFERENCES:	FR009, FR021, FR024
PRIORITY:	++

ID	FR029
TITLE:	End Screen
DESCRIPTION:	When the match finishes, the winning player gets shown on the screen.
EXPLANATION:	The games' outcome should get shown to the user.
REFERENCES:	FR017
PRIORITY:	++

ID	FR030
TITLE:	Match Statistics
DESCRIPTION:	When the match finishes, interesting statistics get shown to the user.
EXPLANATION:	The result of the game should be shown to the user.
REFERENCES:	FR029
PRIORITY:	-

ID	FR031
TITLE:	Replay
DESCRIPTION:	Users can access a replay of a game by getting its log file from the server.
EXPLANATION:	The user might want to watch past games.
REFERENCES:	FR008
PRIORITY:	-

ID	FR032
TITLE:	Requesting Pause
DESCRIPTION:	Players are able to request to pause the match, the server is supposed to hold the game until one of the players request to continue. AI-Clients are not allowed to request pauses or continuation.
EXPLANATION:	Players should be able to pause the game at any time.
REFERENCES:	-
PRIORITY:	++

ID	FR033
TITLE:	Reconnection attempt
DESCRIPTION:	Both user client and AI client shall try upon loss of the TCP connection to the server to reconnect to the server.
EXPLANATION:	Especially for WiFi connections it can happen, that the system is offline for a short time. This should not cause disqualification at any time.
REFERENCES:	FR009, FR021, FR024
PRIORITY:	++

ID	FR034
TITLE:	Change audio settings
DESCRIPTION:	The user has the possibility to change the preset audio settings within the graphical user interface.
EXPLANATION:	To enrich the game experience we decided to add background music to the game, as well as sound effects. Because not every player will want to listen to the music, these settings are provided.
REFERENCES:	-
PRIORITY:	++

ID	FR035
TITLE:	Change username
DESCRIPTION:	A username is needed to connect to a game lobby. This username will be changeable in the main menu settings.
EXPLANATION:	Players shall always be able to see their current username and change it at any time.
REFERENCES:	-
PRIORITY:	++

3.1.3 AI-Client

ID	FR036
TITLE:	Supported operating system
DESCRIPTION:	An AI-Client must be startable from the Linux command prompt using a docker container. <i>Remark: The Standards-committee decides in which form the start parameters shall be passed..</i> When start is completed, no communication takes place between the user and the AI-Client.
EXPLANATION:	A Docker helps making the Application independent from a specific operating system.
REFERENCES:	-
PRIORITY:	++

ID	FR037
TITLE:	Connection type
DESCRIPTION:	AI-Clients can only connect to a game lobby as a player. They send their username, as well the fact, that they are not controlled by a human, to the server.
EXPLANATION:	Differentiating player and AI
REFERENCES:	FR005, FR020
PRIORITY:	++

ID	FR038
TITLE:	Rule compliant AI
DESCRIPTION:	The AI only determines and executes legal actions within the time specified in the match configuration.
EXPLANATION:	A match against an AI has to be fair
REFERENCES:	FR013
PRIORITY:	++

ID	FR039
TITLE:	AI difficulty
DESCRIPTION:	A configuration file or command-line argument can set different difficulties for the AI.
EXPLANATION:	This makes the game more accessible for differently experienced players and challenging at the same time.
REFERENCES:	-
PRIORITY:	-

ID	FR040
TITLE:	AI-Client interface
DESCRIPTION:	The AI-Client provides an interface to be included to the players client.
EXPLANATION:	AI-Client needs to be able to communicate.
REFERENCES:	FR026
PRIORITY:	-

3.1.4 Editor

ID	FR041
TITLE:	JSON format
DESCRIPTION:	he standardization-committee defines JSON schemes for scenarios, match configurations, and character configurations.
EXPLANATION:	Since the system components (Client, Server, AI-Client and Editor) should be combinable across groups, communication must be uniform.
REFERENCES:	-
PRIORITY:	++

ID	FR042
TITLE:	Graphical user interface
DESCRIPTION:	Scenarios, match configurations, and character configurations can be created and edited using a graphical interface.
EXPLANATION:	The user should be provided with a comfortable user interface, to easily create own scenarios and character sets.
REFERENCES:	FR002, FR003, FR004
PRIORITY:	+

ID	FR043
TITLE:	Random scenario
DESCRIPTION:	The editor is capable of generating random scenarios.
EXPLANATION:	This represents a basic functionality of an editor.
REFERENCES:	FR002, FR003, FR004
PRIORITY:	-

ID	FR044
TITLE:	Validation
DESCRIPTION:	The editor can validate user-loaded or created scenarios, match configurations, and character configurations.
EXPLANATION:	Invalid file formatting, possibly leading to a crash is prevented.
REFERENCES:	-
PRIORITY:	++

3.2 Non-functional requirements

ID	QR001
TITLE:	Reliability
DESCRIPTION:	Out of one hundred instances, the system may crash no more than once. A crash occurs when one of the components stops working due to an internal error.
EXPLANATION:	Only a working game can be fun.

ID	QR002
TITLE:	Documentation language
DESCRIPTION:	All documents related to the product as well as the code and its documentation will be written in English.
EXPLANATION:	This way, a country- and language-independent documentation take place.

ID	QR003
TITLE:	Code Documentation
DESCRIPTION:	The entire code gets documented understandably and unambiguously. The coverage should be 95%, whereby direct getters and setters (methods that only encapsulate direct access to a variable), as well as object-derived methods (such as toString()) and automatically generated code, are fundamentally excluded from this statistic.
EXPLANATION:	This guarantees the comprehensibility of the modules even for developers/persons with programming knowledge who were not involved in their development.

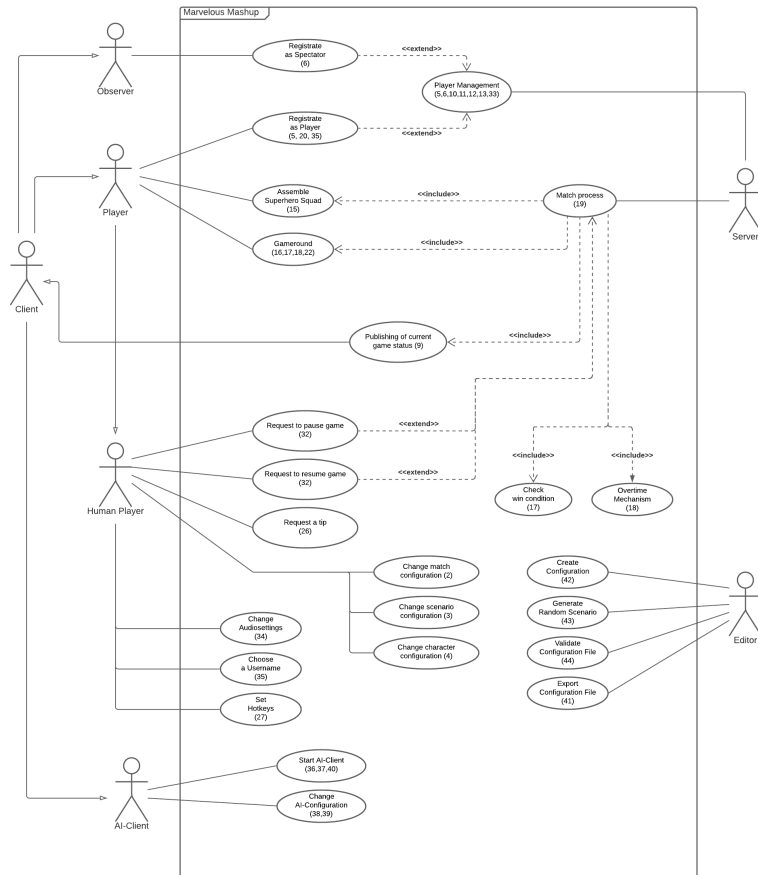
ID	QR004
TITLE:	Supported operating systems
DESCRIPTION:	The components must each run on either Windows, a current Linux distribution, or in a current browser.
EXPLANATION:	This requirement was defined by the customer.

ID	QR005
TITLE:	Testability
DESCRIPTION:	The program components can be tested in the respective unit tests, depending on the language. Furthermore, these tests are 100% successful and cover 75% of the code, excluding automatically generated and GUI specific code.
EXPLANATION:	This way, the program's reliability is ensured.

ID	QR006
TITLE:	Proper network communication
DESCRIPTION:	There may not be a point in time when a syntactically incorrect message is transmitted between client and server.
EXPLANATION:	The components must be able to rely on consistent communication syntax.

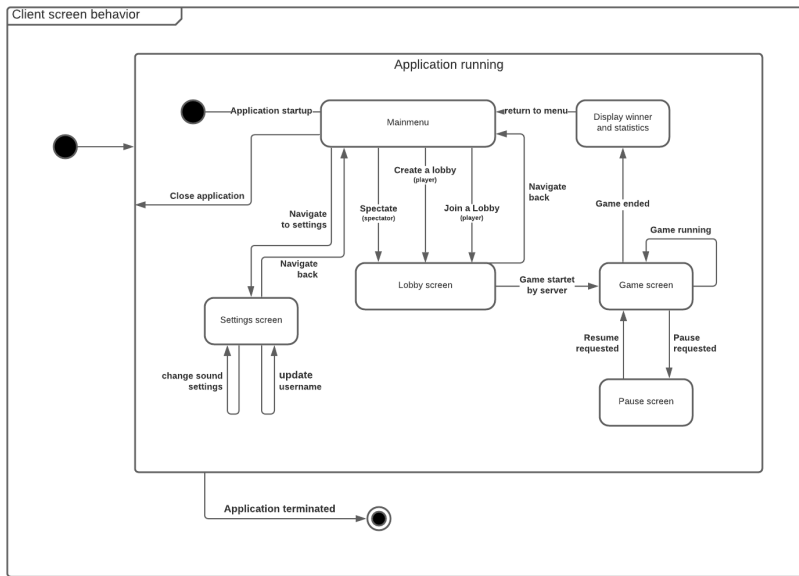
4 Usecases

4.1 Use Case Diagramm

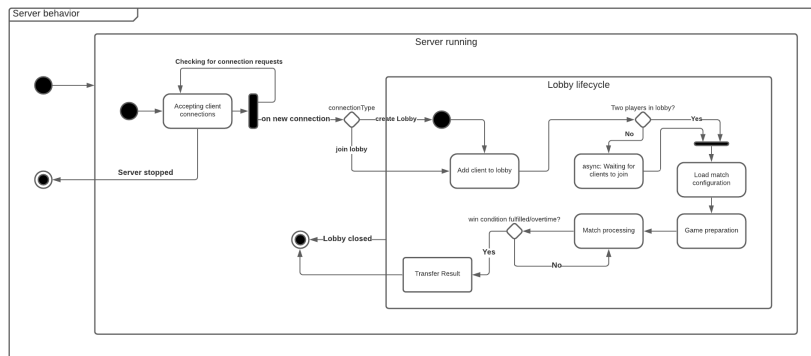


4.2 Component behavior

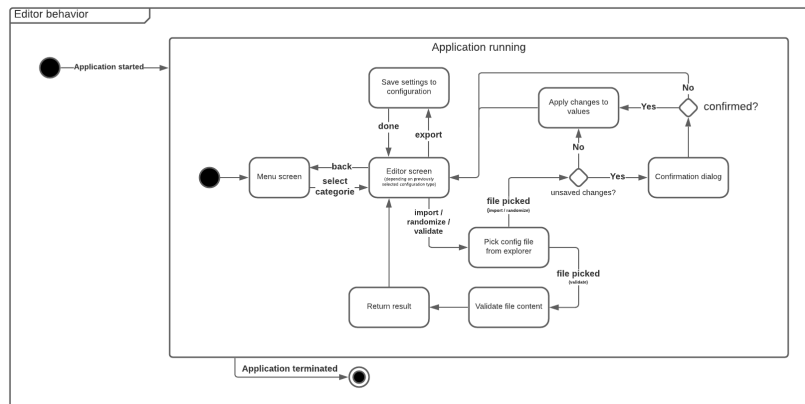
4.2.1 Client



4.2.2 Server

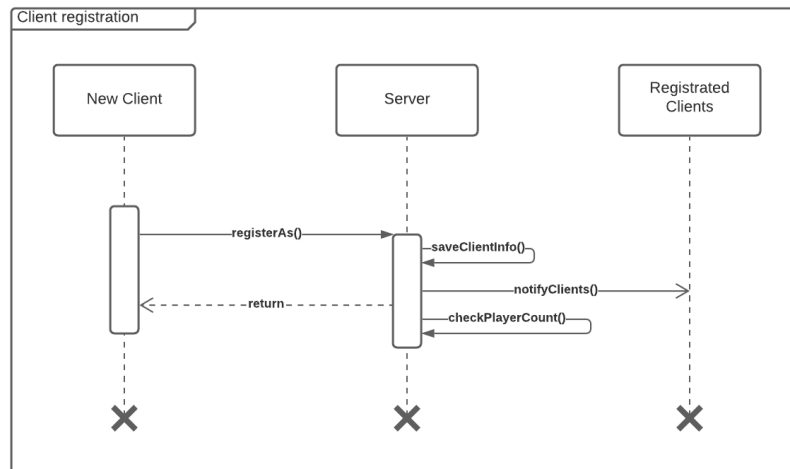


4.2.3 Editor

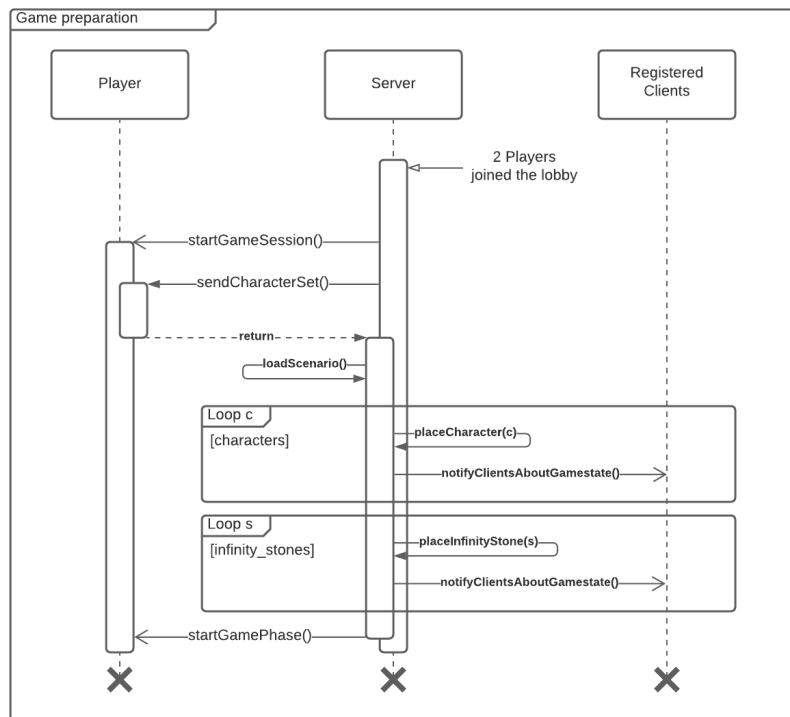


4.3 Gameplay

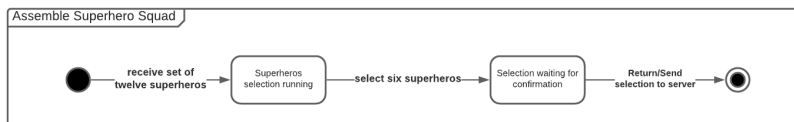
4.3.1 Client registration



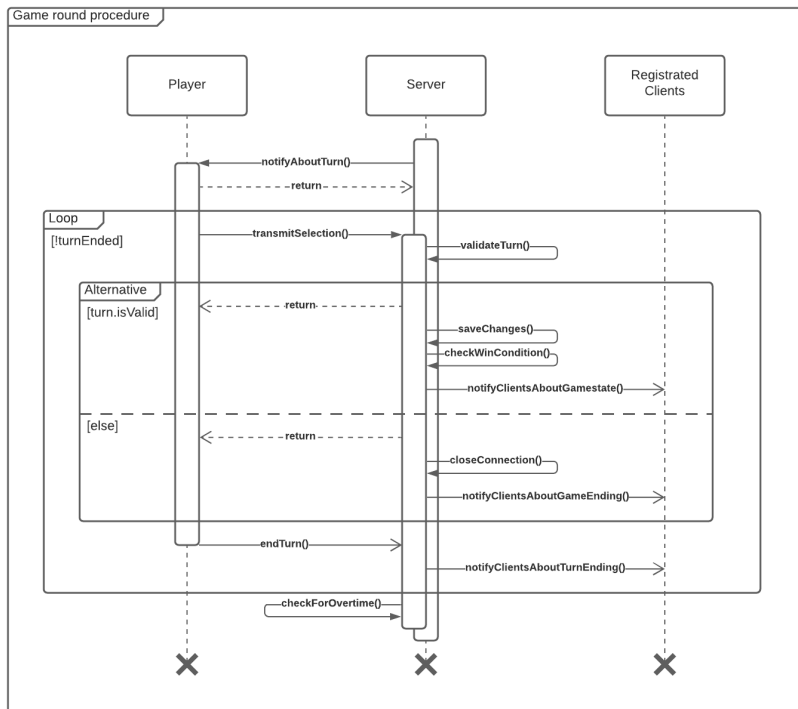
4.3.2 Game preparation



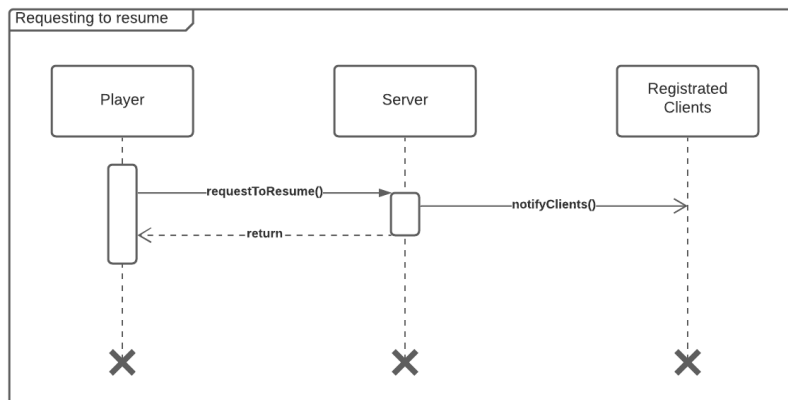
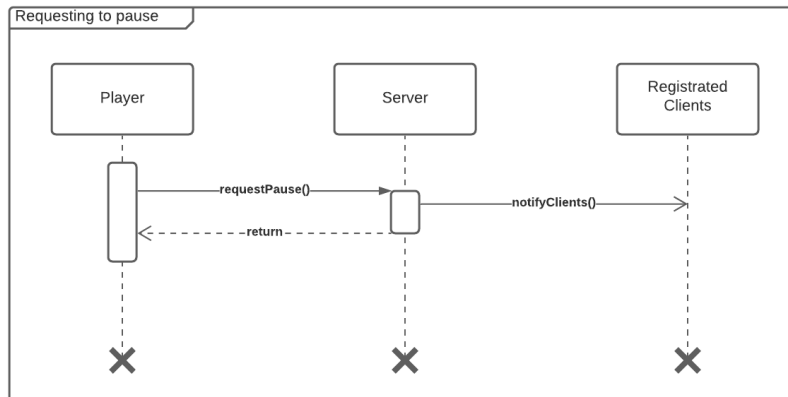
4.3.3 Assemble superhero squad



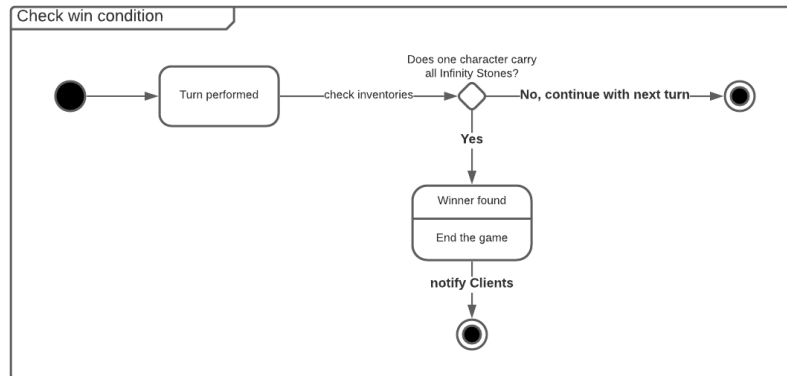
4.3.4 Game round



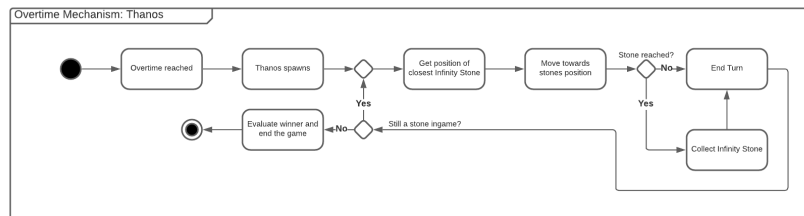
4.3.5 Pause handling



4.3.6 Check win condition

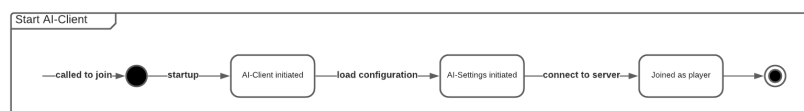


4.3.7 Overtime mechanism



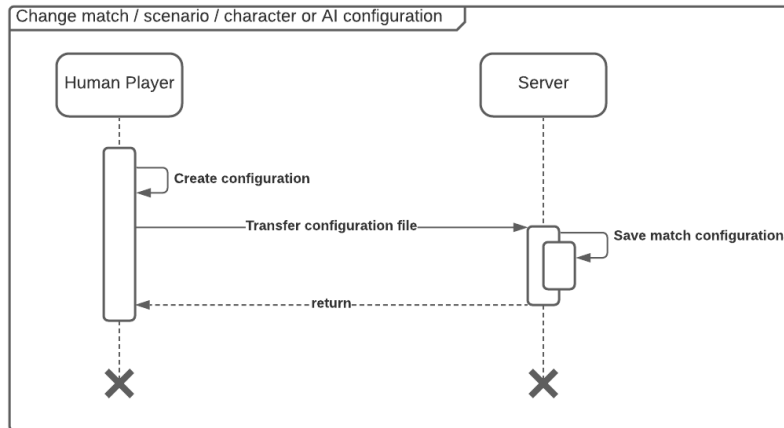
4.4 AI-Client

4.4.1 Startup



4.5 Changing configurations

Uploading a configuration



5 Models and diagrams

In this section, the visualization of the domain model as well as the use cases is presented.

5.1 Domain model

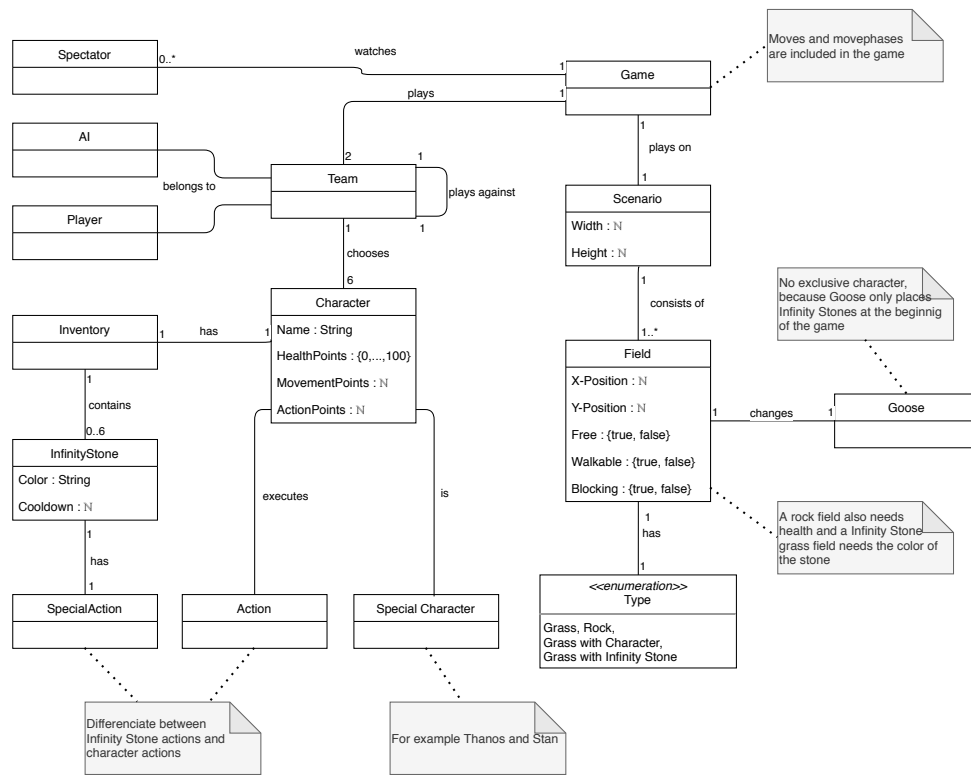


Figure 1: Domain model

5.2 Use case diagram

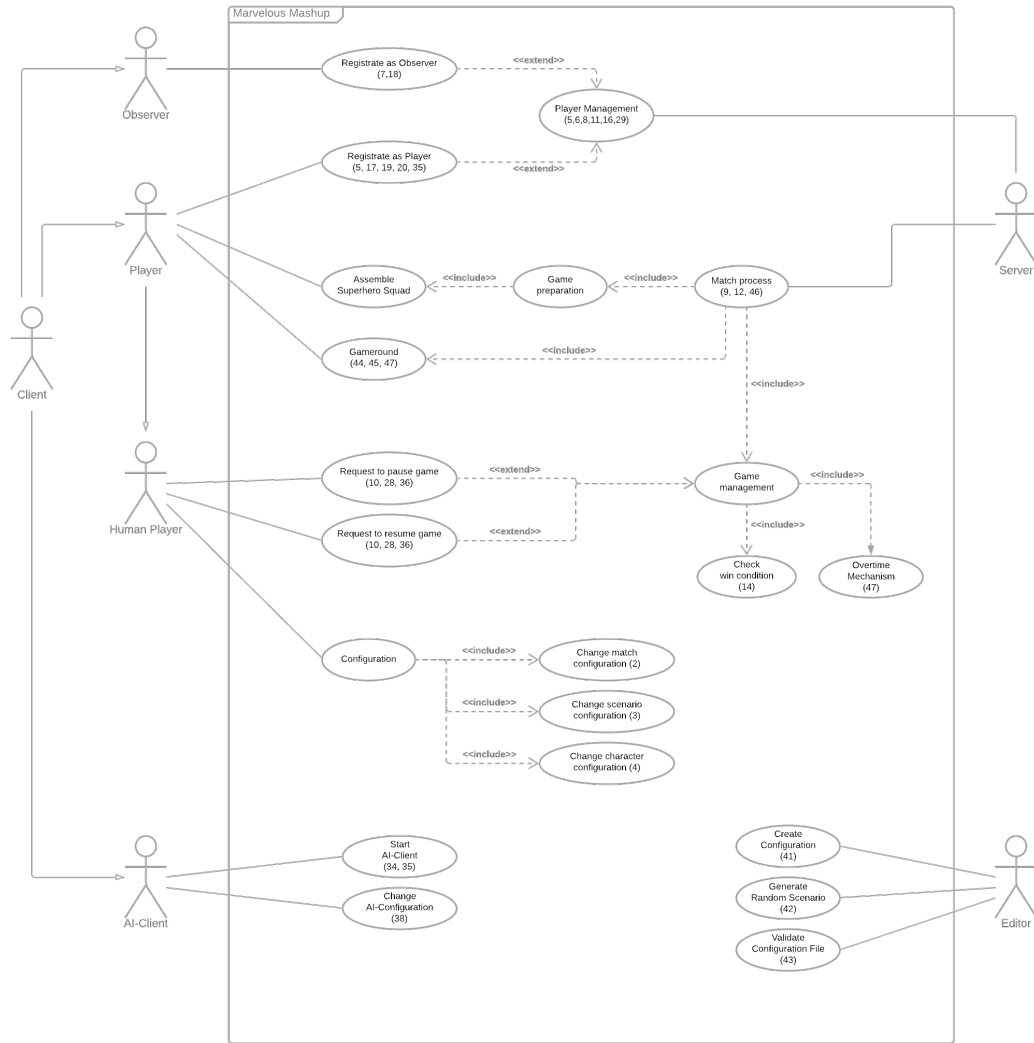


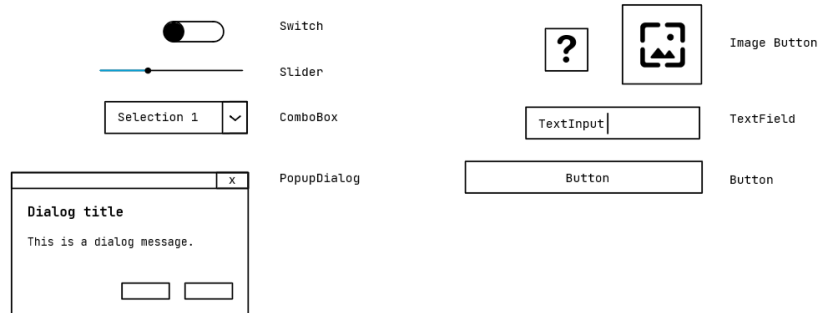
Figure 2: Use case diagram

6 Mock-Ups

In this section, the main graphic design concepts of the individual dialogues are illustrated by mock-up drawings. Textual descriptions also serve to provide a better overview.

6.1 Dialog

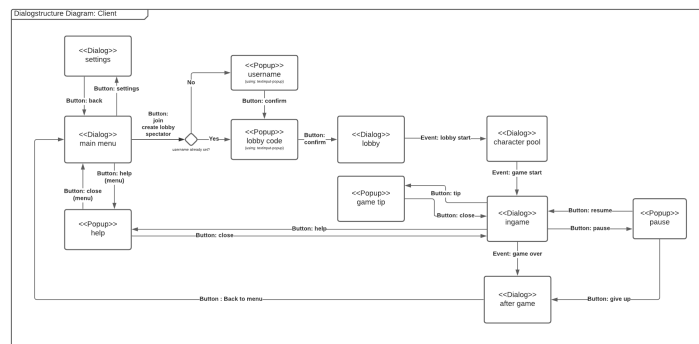
6.1.1 UI Element Overview



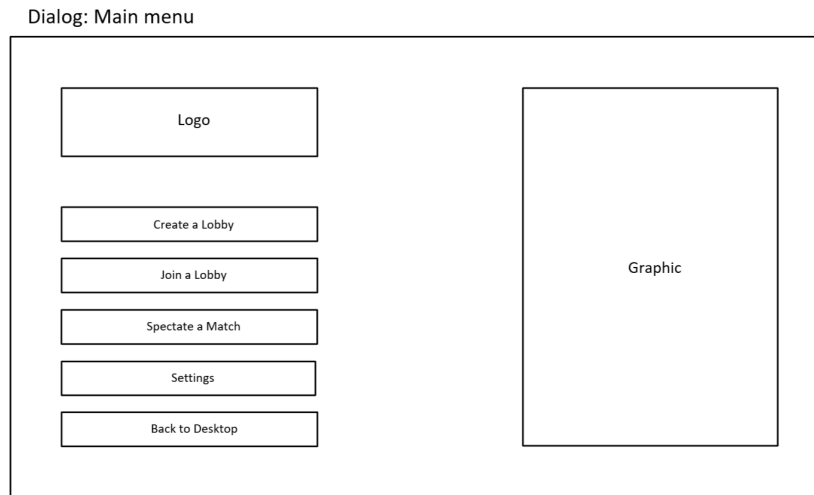
6.1.2 Client

The client component is based on a graphical user interface. This decision was made due to the requirements, which need to be fulfilled. For an easy to overlook and understandable game flow, users are supposed to see the ingame state at every time. In addition to that, navigating within the application is more user friendly by clicking buttons, then console commands which needed to be entered. For the basic design of the client, we developed the following dialog structure and mockups.

6.1.3 Dialog Structure



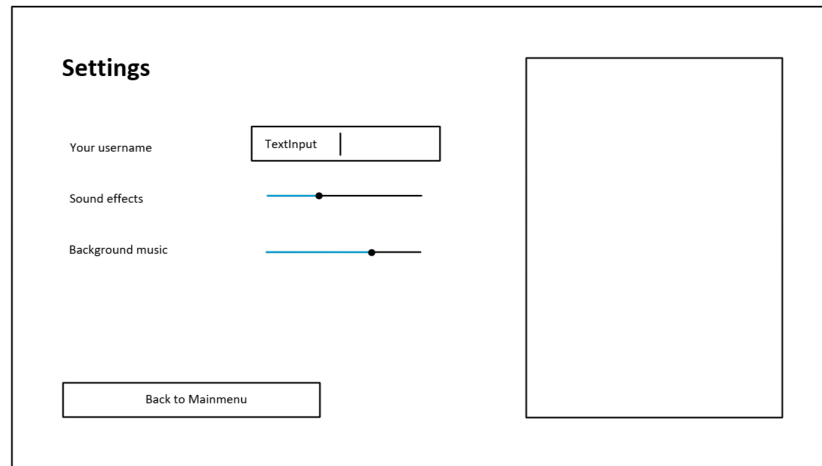
6.1.4 Mockups



Dialog Main Menu

UseCase	Manual
Registrate as player	When pressing "Join Lobby" or "Create Lobby" the Client asks for a name, if it does not have one already (Client settings), and if it is "Join Lobby" the lobby code by opening a TextInput popup. After the client contacts the server to create or join the lobby
Registrate as Spectator	When pressing "Spectate a Match" the client opens a Text Input popup to get the lobby code. After that it connects to the sever as a spectator for this lobby and if the game has not started yet it changes to the Lobby dialog, else it goes to the Ingame dialog.
Change client settings	By pressing "Settings" the client shows the Settings Dialog, where the client's settings can be changed.

Dialog: Settings

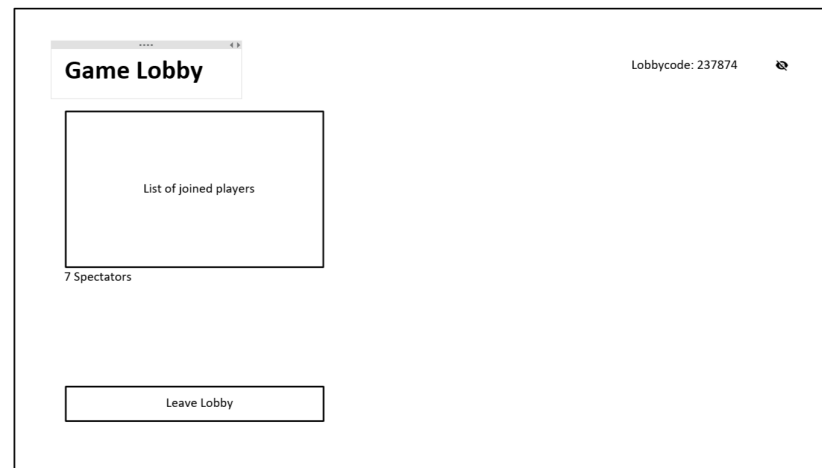


The Settings dialog box has a title bar labeled "Settings". It contains a text input field labeled "Your username" with the placeholder text "TextInput". Below this are two sliders: "Sound effects" and "Background music". At the bottom left is a button labeled "Back to Mainmenu". On the right side of the dialog is a large, empty rectangular area.

Dialog Settings

UseCase	Manual
Change audio settings	Audio settings can be changed by the sliders "Sound effects" and Background music".
Enter a Username	The username can be chosen by entering it in the TextField "Your Username"
Set hotkeys	Hot keys can be chosen by clicking on an action in the list of actions and pressing the specific keys.

Dialog: Lobby

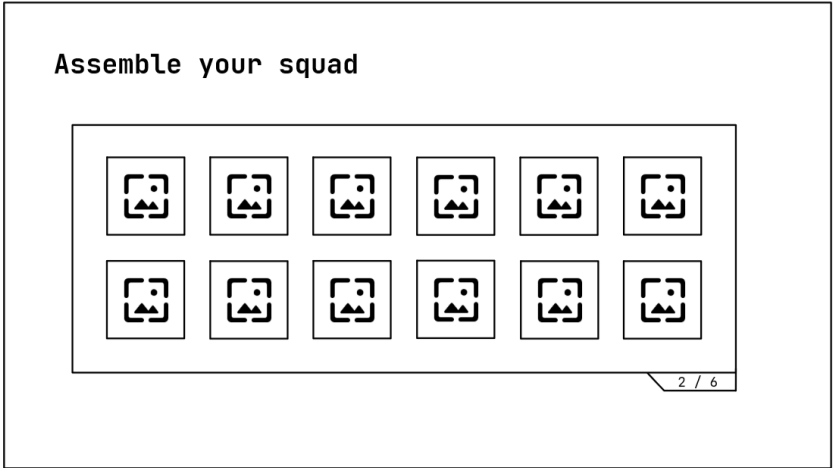


The Game Lobby dialog box has a title bar labeled "Game Lobby". In the top right corner, it displays "Lobbycode: 237874" next to a small icon. Below this is a large rectangular area labeled "List of joined players". Underneath this area, it says "7 Spectators". At the bottom of the dialog is a button labeled "Leave Lobby".

Dialog Lobby Screen

UseCase	Manual
Player management	Client waits for notification from the server that the game starts, which is sent, when there are two players in the game

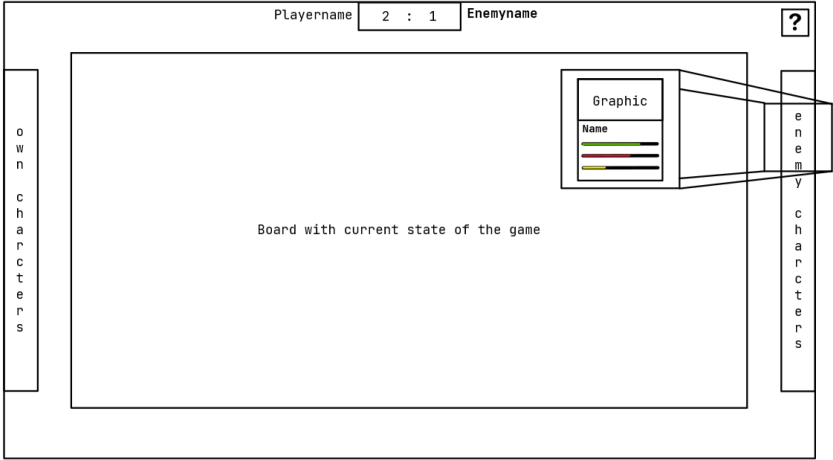
Dialog: Character pool



Dialog Character Pool

UseCase	Manual
Assemble superhero squad	User can select six superheroes from the collection shown for his superhero group. When all six are selected the popup closes automatically

Dialog: Ingame



Dialog Ingame

UseCase	Manual
Request to pause Game	The player can request to pause the game by clicking on the button, in the top right, displaying a pause icon [⏸]. The client then sends a message to the server to request the pause.
Request to resume Game	When receiving a message form the server that the game is paused, the client opens the popup. There the client can request to resume the game by pressing the button "Resume game".
Request a Hint	The user can request a tip by pressing the "light bulb/hint" button, in the bottom right corner. While looking at the rules, the game is paused.
Look at the Rules	The user can look at the rules by pressing the "?/help" button, in the bottom right corner. While looking at the rules, the game is paused.
Game Round	Shows the user the current state and events of the game and gives him the option to choose his actions, when it is his turn.

Dialog: after game

Statistics

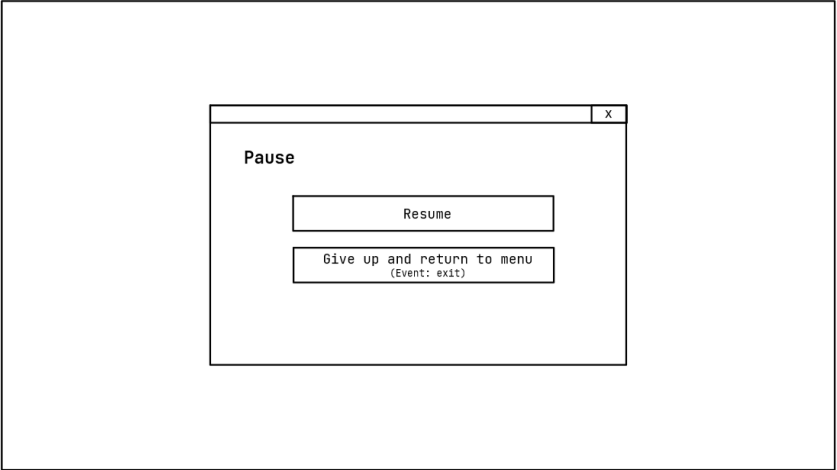
Player 1		Player 2
XX	MP	XX
XX	AP	XX
XX	Actions	XX
XX	Knock-Outs	XX
XX	Attacks	XX
...

Back to main menu

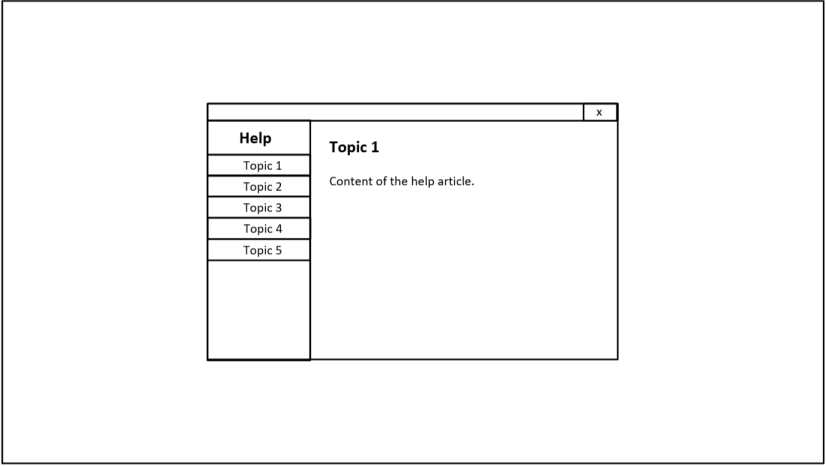
Dialog After Game

UseCase	Manual
Return to menu	User can return to main menu after finishing looking at the statistics.

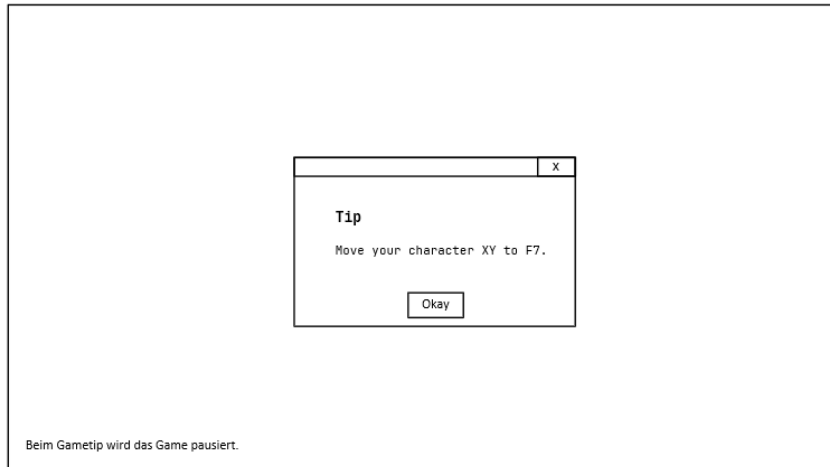
Popup: Pause



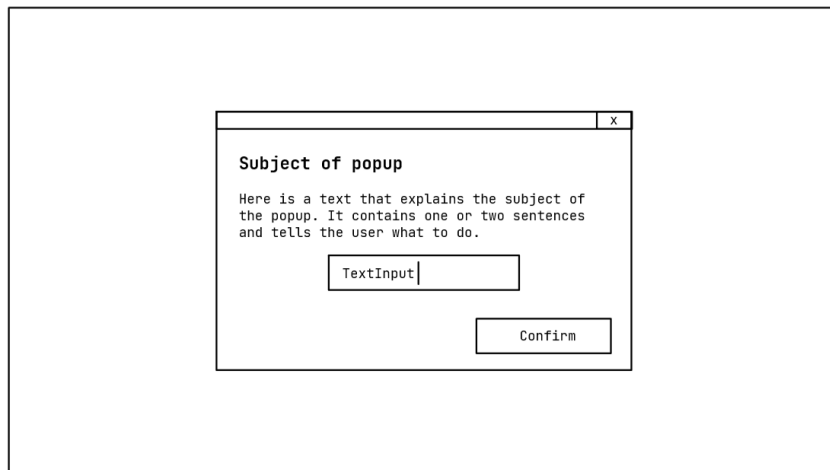
Popup: Help



Popup: Gametip



Popup: TextInput



6.1.5 Editor

The configuration editor is also based on a graphical user interface. The component is meant to be a standalone app that allows users to create their own levels, characters, and match settings. The design idea is to keep the application simple and easy to understand, which lead us to the following application structure and design concepts.

6.1.6 Dialog Structure

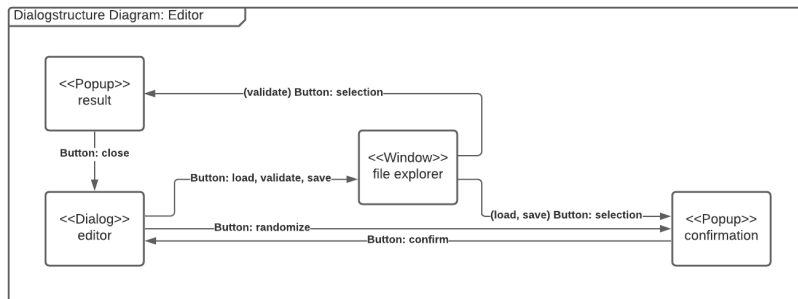
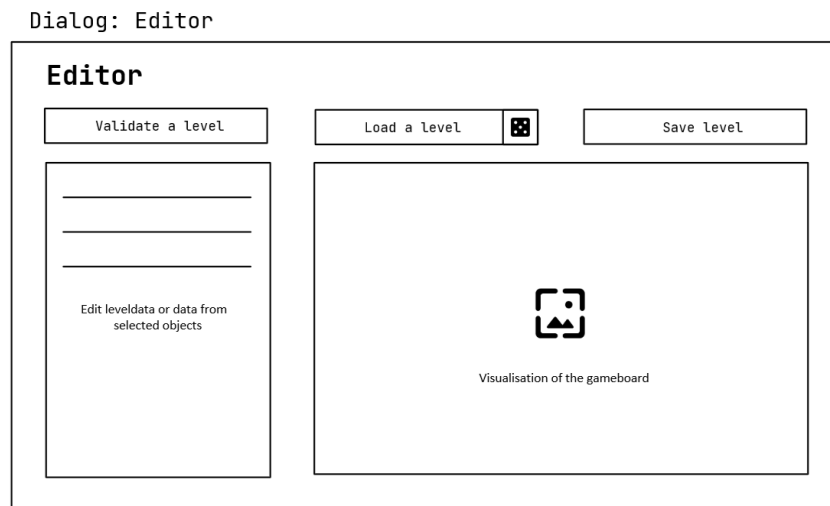


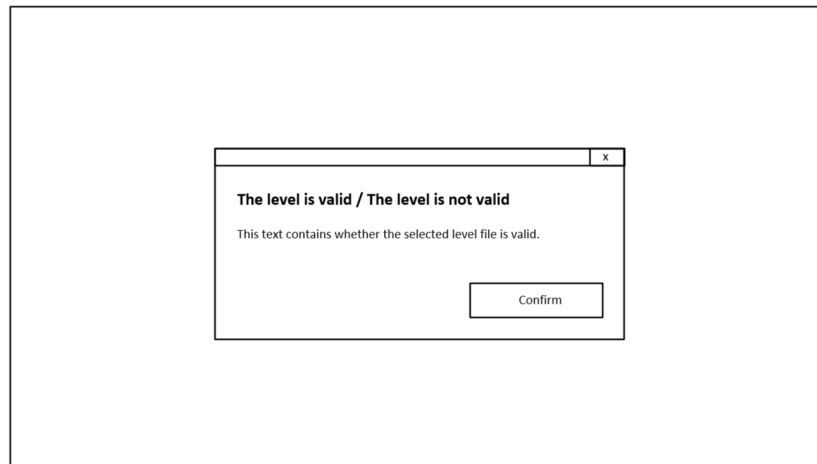
Figure 3: Dialog Structure

6.1.7 Mockups



UseCase	Manual
Create Configuration	<ul style="list-style-type: none"> • The user can load a level by clicking on the "load a level" button. The explorer will be opened in which the user can choose a level file. • In the visualisation of the gameboard the user can see a preview of the final gameboard and manipulate general things about it. The edit screen to it's left is used to fill in more specific details about the level in general and selected objects. • The user can save the current edited level by clicking on the "save level" button. The explorer will be opened in which he can choose a directory and a name for his file.
Generate random Scenario	The user can generate a random level by clicking on the dice button next to "load a level". The values in the editor will be changed to random values.
Validate configuration file	The user can open an explorer window to select a level file that should be validated by clicking on the "validate a level" button. The result is shown in the result popup.

Popup: result



6.1.8 Server

The server component will be implemented with a console window instead of a graphical user interface. For the correct and complete functionality of the application, commands for querying and updating the various configuration files as well as basic system actions will be provided for use.

6.2 Gameboard



Figure 4: no character selected



Figure 5: character selected



Figure 6: character attack



Figure 7: character move

7 Acceptance criteria

In order to create a common basis between the developer and the customer to determine when the project is complete, there are precise acceptance criteria. The acceptance criteria are listed below.

- The editor must be able to create character sets, maps and game configurations, as well as verify existing game configurations.
- The self-developed components (client, editor and AI) must be interchangeable with the components of other teams, without limiting the functionality of the entire project. They must also be compatible with the server of other teams. This ensures compliance with the standardized protocols, the components use to communicate with each other.
- In this requirement specification all functional requirements were listed, which the application must have implemented. These functional requirements are selected in such a way that by adhering to them, the entire game process, as described in the customer's specifications, can be carried out and the user has a functional game in front of him at the end. All functional requirements rated as important must therefore be implemented.
- The Client has a graphical interface for the user, with the help of which all the important functions of the game can be performed. This graphical interface also shows the user the current game state at any given time.