# Specification for Marvelous Mashup

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

December 2020

## 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

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

| **Term** | **Win conditions** |
|---|---|
| DESCRIPTION: | The win conditions are considered in the following order:<br>• 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.<br>• Secondary win condition: The group that possessed the higher amount of Infinity Stones at a certain point in time wins.<br>• Tertiary win condition: The group that knocked out the higher amount of opponent characters wins.<br>• Quaternary win condition: The group that caused the higher amount of Health Point (HP) damage to the opponent team in total wins.<br>• 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: | A character is an object with predefined values set in a character configuration file. These are:<br>• Name: Each character has its name.<br>• Avatar: Each character is represented by an avatar to distinguish between the different participating characters.<br>• 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.<br>• 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.<br>• Inventory: A character can carry Infinity Stones in their inventory. |
| IS-A: | - |
| CAN-BE: | - |
| EXAMPLE: | **Rocket Raccoon:**<br>**HP:** 100<br>**MP:** 2<br>**AP:** 2<br>**Close combat:** Damage 10<br>**Range combat:** Damage 30, Range 5 |

| **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** | **Standardization-Committee** |
|---|---|
| DESCRIPTION: | The standardization-committee is an instance, deciding about communicational conditions, required for implementation. |
| 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 group** |
|---|---|
| DESCRIPTION: | The process of the election phase from a player's point of view is like this:<br>1. The server takes 12 random characters from the character pool and displays them to the player<br>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: | Each field on a board can have 4 states which are defined as follows:<br>• grass: free field, characters can stand on it and move over it.<br>• 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.<br>• 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.<br>• 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: | The following moves can be performed by spending APs:<br>• 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).<br>• 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).<br>• Usage of Infinity Stones: The character uses the special ability from an Infinity Stone in his inventory.<br>• 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: | 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:<br>• Space Stone (blue): The carrier can teleport to any free field.<br>• 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.<br>• 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).<br>• 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.<br>• 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.<br>• 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.<br>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. |
| 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.

| **ID** | **FR001** |
|---|---|
| TITLE: | Starting server |
| DESCRIPTION: | The server must be startable in the Linux command line in form of a Docker-container. ***Remark: The standardization-committee decides in which form the start parameters shall be passed.*** |
| 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 of ?server/match?. |
| EXPLANATION: | The match configuration contains parameters for each match, such as periods allowed per action or the number of rounds until the overtime condition for a match is met. |
| REFERENCES: | |
| PRIORITY: | ++ |

| **ID** | **FR003** |
|---|---|
| TITLE: | Scenario configuration |
| DESCRIPTION: | The server loads a scenario configuration at the start of ?server/match?. |
| EXPLANATION: | The scenario configuration defines, what the game board looks like. |
| REFERENCES: | FR041, FR042 |
| PRIORITY: | ++ |

| **ID** | **FR004** |
|---|---|
| TITLE: | Character configuration |
| DESCRIPTION: | The server loads a character configuration at the start of ?server/match?. |
| EXPLANATION: | The character configuration the pre-defined values for each character. |
| REFERENCES: | FR021, FR042 |
| 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: | FR006, FR017, FR019, FR020, FR034, FR035 |
| PRIORITY: | ++ |

| **ID** | **FR006** |
|---|---|
| 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** | **FR007** |
|---|---|
| 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: | FR018 |
| PRIORITY: | ++ |

| **ID** | **FR008** |
|---|---|
| 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: | FR011 |
| PRIORITY: | ++ |

| **ID** | **FR009** |
|---|---|
| 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** | **FR010** |
|---|---|
| TITLE: | Pause match |
| DESCRIPTION: | If a player requests a pause, 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: | FR028, FR036 |
| PRIORITY: | ++ |

| **ID** | **FR011** |
|---|---|
| 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: | FR008, FR029 |
| PRIORITY: | ++ |

| **ID** | **FR012** |
|---|---|
| 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, FR007, FR009, FR037 |
| PRIORITY: | ++ |

| **ID** | **FR013** |
|---|---|
| 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, FR023, FR027 |
| PRIORITY: | ++ |

| **ID** | **FR014** |
|---|---|
| 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: | FR030 |
| PRIORITY: | ++ |

| **ID** | **FR015** |
|---|---|
| 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: | FR032 |
| PRIORITY: | - |

| **ID** | **FR016** |
|---|---|
| TITLE: | Web Socket |
| DESCRIPTION: | The port for the web socket connection should be 1218. |
| EXPLANATION: | That's us. |
| REFERENCES: | FR017, FR018, FR019, FR020 |
| PRIORITY: | - |

| **ID** | **FR017** |
|---|---|
| TITLE: | Player registration |
| DESCRIPTION: | A user can register as a player for a match. Therefore he uses the network of a server to enroll. |
| EXPLANATION: | Players have to enter the game to start a match. |
| REFERENCES: | FR005, FR016, |
| PRIORITY: | ++ |

| **ID** | **FR018** |
|---|---|
| TITLE: | View mode |
| DESCRIPTION: | The users have a view-mode to act as a spectator. They can enter this state any time of a match by registering to the server. |
| EXPLANATION: | The server has to distinguish between player and spectator. |
| REFERENCES: | FR007, FR016 |
| PRIORITY: | ++ |

| **ID** | **FR019** |
|---|---|
| TITLE: | User Client |
| DESCRIPTION: | The user is controlled by a human and therefore informs the server. |
| EXPLANATION: | The server is supposed to know if a client is controlled by a human or an AI. |
| REFERENCES: | FR005, FR016 |
| PRIORITY: | ++ |

| **ID** | **FR020** |
|---|---|
| TITLE: | Name delivery |
| DESCRIPTION: | The clients transmit their name to the server. |
| EXPLANATION: | Spectators can distinguish between the participating players. |
| REFERENCES: | FR005, FR016 |
| PRIORITY: | ++ |

| **ID** | **FR021** |
|---|---|
| TITLE: | Graphical 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: | FR004, FR013, FR022, FR023, FR027, FR033 |
| PRIORITY: | ++ |

| **ID** | **FR022** |
|---|---|
| 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, FR023 |
| PRIORITY: | ++ |

| **ID** | **FR023** |
|---|---|
| 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: | FR013, FR021, FR022, FR027 |
| PRIORITY: | ++ |

| **ID** | **FR024** |
|---|---|
| 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: | FR012 |
| PRIORITY: | ++ |

| **ID** | **FR025** |
|---|---|
| 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: | FR039 |
| PRIORITY: | – |

| **ID** | **FR026** |
|---|---|
| TITLE: | Operating features |
| 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** | **FR027** |
|---|---|
| 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: | FR013, FR021, FR023 |
| PRIORITY: | ++ |

| **ID** | **FR028** |
|---|---|
| TITLE: | Break Client |
| DESCRIPTION: | The user client allows the user to make a wish for a break. If the game is currently paused, the user shall be able to make request for the game to be continued. Both shouldn't be possible for AI clients. |
| EXPLANATION: | The user should be able to pause and unpause the game. |
| REFERENCES: | FR010 |
| PRIORITY: | ++ |

| **ID** | **FR029** |
|---|---|
| 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. That shouldn't cause disqualification every time. |
| REFERENCES: | FR011 |
| PRIORITY: | ++ |

| **ID** | **FR030** |
|---|---|
| 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: | FR014 |
| PRIORITY: | ++ |

| **ID** | **FR031** |
|---|---|
| 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: | FR030 |
| PRIORITY: | - |

| **ID** | **FR032** |
|---|---|
| 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: | FR015 |
| PRIORITY: | - |

| **ID** | **FR033** |
|---|---|
| TITLE: | Stan Lee Easter Egg |
| DESCRIPTION: | There's a Stan Lee Easter-egg hidden on the graphical interface. |
| EXPLANATION: | Because that's how it's supposed to be. |
| REFERENCES: | FR021 |
| PRIORITY: | - |

| **ID** | **FR034** |
|---|---|
| TITLE: | Starting AI-Client |
| DESCRIPTION: | An AI-Client must be startable from the Linux command prompt using a docker container. ***Remark: The Standards-committee decides in which form the start parameters shall be passed.***. 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: | FR005 |
| PRIORITY: | ++ |

| **ID** | **FR035** |
|---|---|
| TITLE: | AI Server notification |
| DESCRIPTION: | AI notifies the server that it is an AI. |
| EXPLANATION: | Differentiating player and AI |
| REFERENCES: | FR005 |
| PRIORITY: | ++ |

| **ID** | **FR036** |
|---|---|
| TITLE: | Pause |
| DESCRIPTION: | An AI must be able to deal with the pause of a player. It cannot initiate a pause itself. |
| EXPLANATION: | The player should decide when to pause |
| REFERENCES: | FR010 |
| PRIORITY: | + |

| **ID** | **FR037** |
|---|---|
| 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: | FR012 |
| PRIORITY: | ++ |

| **ID** | **FR038** |
|---|---|
| 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** | **FR039** |
|---|---|
| TITLE: | AI-Client interface |
| DESCRIPTION: | The AI-Client provides an interface to be included to the players client. |
| EXPLANATION: | The AI-Client can then be used to show recommended moves to the player. |
| REFERENCES: | FR025 |
| PRIORITY: | – |

| **ID** | **FR040** |
|---|---|
| TITLE: | JSON format |
| DESCRIPTION: | The 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** | **FR041** |
|---|---|
| TITLE: | Editor |
| 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 so that own scenarios can be created easily |
| REFERENCES: | FR003, FR042 |
| PRIORITY: | + |

| **ID** | **FR042** |
|---|---|
| TITLE: | Random scenario |
| DESCRIPTION: | The editor is capable of generating random scenarios. |
| EXPLANATION: | This represents a basic functionality of an editor. |
| REFERENCES: | FR003, FR004, FR041 |
| PRIORITY: | – |

| **ID** | **FR043** |
|---|---|
| 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: | – |

| **ID** | **FR044** |
|---|---|
| TITLE: | Movement |
| DESCRIPTION: | A character can move to an empty field, a field with another character, or a field with an infinity stone next to its location, as long as the character has MPs left. If another character is already on the field, the characters swap fields. If the field contains an infinity stone, it gets placed in the inventory of the character. One MP is then deducted for this round. |
| EXPLANATION: | Fundamental component of the game |
| REFERENCES: | |
| PRIORITY: | ++ |

| **ID** | **FR045** |
|---|---|
| TITLE: | Action |
| DESCRIPTION: | In a movement phase, the player can decide to move his character or perform an action. |
| EXPLANATION: | Fundamental component of the game. |
| REFERENCES: | |
| PRIORITY: | ++ |

| **ID** | **FR046** |
|---|---|
| TITLE: | Match process |
| DESCRIPTION: | 1. Intro spot<br>2. Assemble superhero group<br>3. Character placement<br>4. Goose Rounds (Rounds 1-6)<br>5. Stan Lee Round<br>6. Rounds |
| EXPLANATION: | That's how a match is supposed to work. |
| REFERENCES: | |
| PRIORITY: | ++ |

| **ID** | **FR047** |
|---|---|
| TITLE: | Overtime mechanism : Thanos |
| DESCRIPTION: | 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.<br>• Thanos ignores obstacles. He can move through rocks which then crumble and turn to grass fields<br>• 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<br>• 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<br>• Thanos is immune to all attacks and can't be knocked-out<br>• Characters can't move to a field Thanos is on<br>This is repeated until Thanos possesses all Infinity Stones. Meanwhile, his MPs are increased by one with every turn. |
| EXPLANATION: | Accelerates game to come to an end after set #MAX_ROUND. |
| 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 Models

## 4.1 Domain model

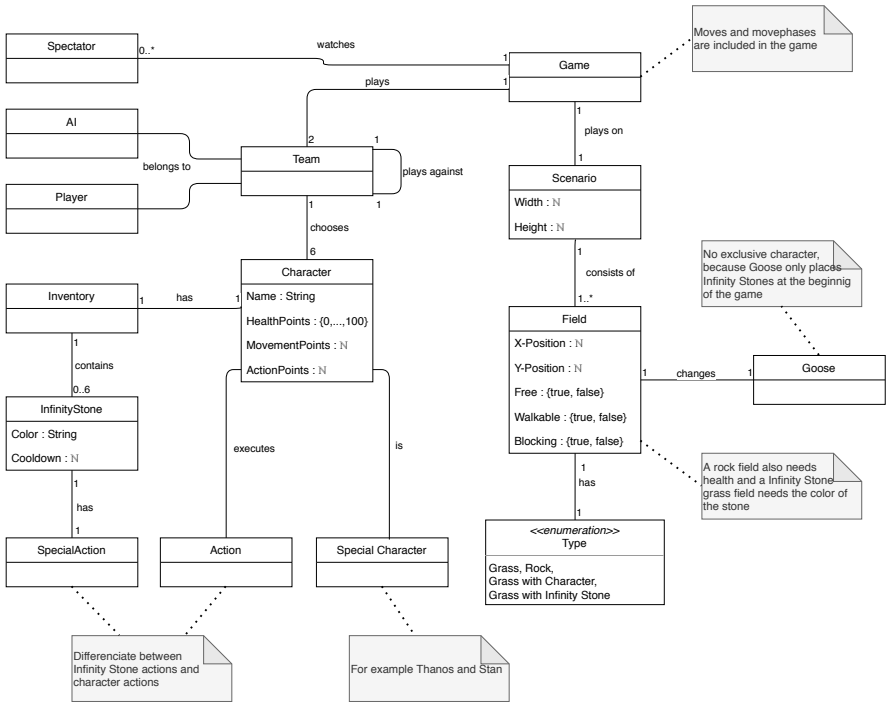


Figure 1: Domain model