Engineering Method
Application for solving a problem
Jhonatan Alexis Arenas Bonilla

Contenido

Phase 1. Problems identify:

Problem definition:

Causes and Symptoms:

The principal symptom of the problem is that, due of the many suggestions received by players in the last months, made Epic Games.inc to be disquiet. The reason for this, is based on reports where the players express some of their inconvenience with the gameplay in their matches.

Regarding previous petitions, derives three problems:

1) There are a difficult by users to find balanced matches, that does not allow them to compete against people who have the same abilities.
2) Absence of mechanism that allow players to face rivals that have the same platform.
3) Scarcity of variety of game modes to celebrate special occasions.

Problem's definition:

Epic games.inc wants a proposal on how to implement new features that allow satisfying requests to their product.

Requirements:

Functional Requirements:

Name	Rating System	Number #	1
Description	Allows to classify players and match them in balanced games where players have the most similar abilities possible and equals physicals conditions. This last part means pings and geolocation.		
Input	Players profile History of players games. Players ping		
Output	Balanced Queues where optimal and equals conditions are for each player.		

Name	Rank player	Number #	1.1
Description	Allows to define in a standard classification the ability of a given player		
Input	Player profile History of player's games.		

	Standard
Output	A ranked player

Name	Compare Players	**Number #**	1.2.1
Description	Mechanism that allows to assert whether a player is betters, or has a better network condition, than other. Otherwise, conclude that their abilities and physical condition (Network) are near or equals.		
Input	Players rank Player Geolocation Players Ping Standard		
Output	An assert that provide information about conditions equality.		

Name	Rank compare	**Number #**	1.2.1.1
Description	Mechanism that allows to assert whether a player is betters than other in ranking approach. Otherwise, conclude that their abilities are near or equals.		
Input	Players rank Standard		
Output	An assert that provide information about Rank equality.		

Name	Latency compare	**Number #**	1.2.1.2
Description	Mechanism that allows calculate ping's standard deviation for a set of players.		
Input	Set of players Player Geolocation		

	Players Ping		
	Standard		
Output	Give standard deviation for a set of players.		

Name	Match Players	**Number #**	1.2.2
Description	Mechanism that allows match players that are searching for a game. That mechanism makes sure to unite players with skills and network condition nears or equals, that do not put them at a disadvantage.		
Input	Conditions equality Set of players		
Output	A balanced queue.		

Name	Platform mode	**Number #**	2
Description	Allows players enqueuer in matches where their face rivals with the same platform.		
Input	Players platform		
Output	Special queue where ranking system are included but a only platform is accepted.		

Name	Saint Valentine mode	**Number #**	3
Description	A new mod where players could only use the last weapon that they have raised.		
Input	Players weapons. Weapons ammunition.		
Output	Special system that deny change weapon until actual weapon has no ammunition.		

No Functional Requirements:

Name	Average wait time	**Number #**	1

Description	Ranking system won't increase the average waiting time.
Input	
Output	An efficient mechanism that provide a rank system without an increase on the waiting time.

Name	Players ranking update	Number #	2
Description	After a match, system should update player's information based on his/her performance in the match.		
Input			
Output	Players rank constantly updated.		

Name	Data structures	Number #	3
Description	Implement known data structure.		
Input			
Output			

Phase 2. Information Gathering:

Unknown words and concepts:

Ranking:

A ranking is a relationship between a set of items such that, for any two items, the first is either 'ranked higher than', 'ranked lower than' or 'ranked equal to' the second. In mathematics, this is known as a weak order or total preorder of objects. It is not necessarily a total order of objects because two different objects can have the same ranking. The rankings themselves are totally ordered.

Latency:

Latency is a networking term to describe the total time it takes a data packet to travel from one node to another. Latency refers to time interval or delay when a system component is waiting for another system component to do something. This duration of time is called *Latency.*

In online gaming, lag is noticeable delay between the action of players and the reaction of the server in a videogame.

Ping refers to the network latency between a player's client and the game server as measured with the ping utility or equivalent. Ping is reported quantitatively as an average time in milliseconds (ms). *High ping* and *low ping* are commonly used terms in online gaming, where high ping refers to a ping that causes a severe amount of lag; while any level of ping may cause lag, severe lag is usually caused by a ping of over 100 ms.

But! This usage is a gaming cultural colloquialism and is not commonly found or used in professional computer networking circles.

Related elements:

Ranking in video games:

Lately, on the Massive Multiplayers Online (MMO) video games appears ranking concept. That respond the necessity to classify every player in the game. Thus, the video game be involved in a new world: "Electronic-sports". That is the case of Dofus[1] where they're going to implement a rank system. Generally, ranking system is based in an index which represent player's performance. That index is called ELO.

Often, MMO implement leagues. That league provides an aesthetic way to view ELO. The most frequent way to implement league system is the next: Create league with the followings name that represent a pyramid hierarchy: bronze, silver, gold, platinum and diamond, by this point some games changes their own numbers to represent places where there only the best players. For instance, League of legends has Master and Challenger leagues. At each league Elo's threshold are defined, when a player improves and cross threshold of him/her actual league, gets promoted to the next league. Some games, also, implement divisions system tan only forward players to improve with small progress.

Nevertheless, each rank system function by points that players can earn or miss in each match. Also, rank system divides normal queues from rank queues.

---

[1] A Strategical MMO.

"The **Elo rating system** is a method for calculating the relative skill levels of players in zero-sum games such as chess. It is named after its creator Arpad Elo, a Hungarian-American physics professor."

"A player's Elo rating is represented by a number which increases or decreases depending on the outcome of games between rated players. After every game, the winning player takes points from the losing one. The difference between the ratings of the winner and loser determines the total number of points gained or lost after a game. In a series of games between a high-rated player and a low-rated player, the high-rated player is expected to score more wins. If the high-rated player wins, then only a few rating points will be taken from the low-rated player. However, if the lower-rated player scores an upset win, many rating points will be transferred. The lower-rated player will also gain a few points from the higher rated player in the event of a draw. This means that this rating system is self-correcting. A player whose rating is too low should, in the long run, do better than the rating system predicts, and thus gain rating points until the rating reflects their true playing strength."

normal distribution:

Related alternatives:

TrueSkill is a skill-based ranking system developed by Microsoft for use with video game matchmaking on Xbox Live. Unlike the popular Elo rating system, which was initially designed for chess, TrueSkill is designed to support games with more than two players.

A player's skill is represented as a *normal distribution "N"* characterized by a mean value of μ (**Mu**, representing perceived skill) and a variance of σ (**Sigma**, representing how "unconfident" the system is in the player's **μ** value).

Player ranks are displayed as the conservative estimate of their skill, **R = μ-3*σ**. This is conservative, because the system is 99% sure that the player's skill is higher than what is displayed as their rank.

The system can be used with arbitrary scales, but Microsoft uses a scale from 0 to 50 for Xbox Live. Hence, players start with a rank of **R = 25-3*25/3 = 0**. This means that a new player's defeat results in a large sigma loss, which partially or completely compensates their mu loss. This explains why people may gain ranks from losses.

Elo calculation:

A player's *expected score* is their probability of winning plus half their probability of drawing. Thus, an expected score of 0.75 could represent a 75% chance of winning, 25% chance of losing, and 0% chance of drawing. On the other extreme it could represent a 50% chance of winning, 0% chance of losing, and 50% chance of drawing. The probability of drawing, as opposed to having a decisive result, is not specified in the Elo system. Instead a draw is considered half a win and half a loss.

If Player A has a rating of **Ra** and Player B a rating of **Rb**, the exact formula (using the logistic curve) for the expected score of Player A is:

$$E_a = \frac{1}{1 + 10^{(\frac{R_b - R_a}{400})}}$$

Similarly, the expected score for Player B is:

$$E_a = \frac{1}{1 + 10^{(\frac{R_a - R_b}{400})}}$$

When a player's actual tournament scores exceed their expected scores, the Elo system takes this as evidence that player's rating is too low and needs to be adjusted upward. Similarly, when a player's actual tournament scores fall short of their expected scores, that player's rating is adjusted downward. Elo's original suggestion, which is still widely used, was a simple linear adjustment proportional to the amount by which a player over performed or underperformed their expected score. The maximum possible adjustment per game, called the **K-factor**, was set at *K* = 16 for masters and *K* = 32 for weaker players.

Supposing Player A was expected to score E_A points but actually scored S_A points. The formula for updating their rating is:

$$R'_A = R_A + K(S_A - E_A)$$

This update can be performed after each game or each tournament, or after any suitable rating period. Suppose Player A has a rating of 1613 and plays in a five-round tournament. He or she loses to a player rated 1609, draws with a player rated 1477, defeats a player rated 1388, defeats

a player rated 1586, and loses to a player rated 1720. The player's actual score is (0 + 0.5 + 1 + 1 + 0) = 2.5. The expected score, calculated according to the formula above, was (0.51 + 0.69 + 0.79 + 0.54 + 0.35) = 2.88. Therefore, the player's new rating is (1613 + 32(2.5 − 2.88)) = 1601, assuming that a K-factor of 32 is used. Equivalently, each game the player can be said to have put an ante of K times their score for the game into a pot, the opposing player also puts K times their score into the pot, and the winner collects the full pot of value K; in the event of a draw the players split the pot and receive K/2 points each.

Note that while two wins, two losses, and one draw may seem like a par score, it is worse than expected for Player A because his or her opponents were lower rated on average. Therefore, Player A is slightly penalized. If Player A had scored two wins, one loss, and two draws, for a total score of three points, that would have been slightly better than expected, and the player's new rating would have been (1613 + 32(3 − 2.88)) = 1617.

Phase 3. Searching for creative solutions:

Idea generation techniques:
I based my technique in "Brainstorming" and create a dynamic that allow me to have a wider view about how the problem can be solved. I will explain in the next paragraph.

For each requirement I will following next algorithm: First, I will start to imagine how can I satisfice it. So then, I wonder if a structure and algorithm could finish with the problem. If in determinate moment, I imagine some different way I will write it in a piece of paper. Once I have done with my first idea, I start to develop that idea or ideas that were presented a few moments ago. If there were presented an idea while I was developing a concept I will repeat that process.

 When I star working on a new requirement, new ideas will be based in previous alternatives. But if, again, there were presented a new concept about how to implement a solve for this requirement, I will use my algorithm: write it, finish to develop actual concept, and star develop new concept. Moreover, this new concept will be a new alternative even I need to change parts in previous requirements.

Alternatives ideas:
Ranting system:

1) Elo implementation:

Each player has a personal score that represent their abilities called *"elo"*. When a new player is registered on Fornite their elo is cero and it cannot decrease more than cero. But as they play matches, their score will progress with.

To compare players, the system proceeds to calculate the expected score by each player and compares these scores. Whether the difference of their possibilities to win are less than 10, both players have almost equal level. Else, player which has the most chance to win will be marked as better. This process is called rank compare.

To compare players' latency, each player will have an attribute with their ping that means their connection status. Moreover, the system proceeds to calculate whether both players have the same ping o almost equals. Based in a difference threshold, the system compares both player, whether difference is above that threshold, the player which has less ping than other will be marked as in better connection status. This process is called latency compare.

So then, to compare players, the system, based on results of ranked compare and latency compare define each player as in same or equals conditions whether: both players have the same or almost equals abilities and connection status. If some player has better possibilities to win and a better connexon status, compare players will notice that both players are not in same conditions.

Players that want to play are enqueuering in a priority queue, where priority will be the searching time for a match.

Keeping match is a list where players are enquired keep for list filling. The system will have a limit of keeping match list. So then, when some of them will filled, it will be sending to start a match.

Each list has with an interval that define its required latency and ability to add some player. That interval should be given by the first player that was added.

When a player wants to be add in a keeping match list, the system proceeds to compare him/her condition with the first player added. Whether player that wants to be add is considered as same or almost equals than the first player so will be add. In case that player will be rejected in each list, but system has a free list, player will be the first to be added in that free list.

Once match has started, the rating system will be in charge to control players elo points. For this, matches will have a table where add the players, given by keeping match list, and identify which player is die or live.

When a player dies, or win, the system proceeds to recalculate her/him elo points with the next way:

- Systems assign that player as winner between players that died before than him/her and loser front players that are alive until that moment. So then, player will be marked as defeat in the previous table.
- With help of the elo function, system define $R_A = player's\ elo$; $R_B = media's\ elo$; $E_A =$ sum of the expected scores against each play (calculate by $\frac{1}{1+10^{\frac{R_B-R_A}{400}}}$, but R_B correspond to each player in the match)
- Once done that calculate, proceeds to add one by each player that dies before than him/her, this would be S_A result.
- Next, the new player's elo is defined by the next formula: $R'_A = R_A + K(S_A - E_A)$

K factor is defined by the next way:

- K=32 to player which has less than 2100 elo points
- K=24 to player which has between 2100 and 2400 elo points
- K=16 to player which has more than 2400 elo points

1) Elo implementation variant 1:

Each player has a personal score that represent their abilities called *"elo"*. When a new player is registered on Fornite their elo is cero and it cannot decrease more than cero. But as they play matches, their score will progress with.

To compare players, the system proceeds to calculate the expected score by each player and compares these scores. Whether the difference of their possibilities to win are less than 10, both players have almost equal level. Else, player which has the most chance to win will be marked as better. That difference threshold will be defined according the situation. This process is called rank compare

To compare players' latency, each player will have an attribute with their ping that means their connection status. Moreover, the system proceeds to calculate whether both players have the same ping o almost equals. Based in a difference threshold, the system compares both player, whether difference is above that threshold, the player which has less ping than other will be marked as in better connexon status. That difference threshold will be defined according the situation. This process is called latency compare.

So then, to compare players, the system, based on results of ranked compare and latency compare define each player as in same or equals conditions whether: both players have the same or almost equals abilities and connexon status. If some player has better possibilities to win and a better connexon status, compare players will notice that both players are not in same conditions. In this step, the system will provide rank compare and latency compare a threshold.

Players that want to play are enqueueing in a priority queue, where priority will be the searching time for a match.

Keeping match is a list where players enquired keep for list filling. The system will have a limit of keeping match list. So then, when some of them will filled, it will be sending to start a match.

Each list has with an interval that define its required latency and ability to add some player. That interval should be given by the first player that was added.

When a player wants to be add in a keeping match list, the system proceeds to compare him/her condition with the first player added. Whether player that wants to be add is considered as same or almost equals than the first player so will be add. In case that player will be rejected in each list, but system has a free list, player will be the first to be added in that free list.

If a player does not be added in a long time, the system determinate which queue has the same conditions that the player and allow that player be added in that list. In the previous case, the system creates a "exception" enlarging the list's interval to compare.

Once match has started, the rating system will be in charge to control players elo points. For this, matches will have a table where add the players, given by keeping match list, and identify which player is die or live.

When a player dies, or win, the system proceeds to recalculate her/him elo points with the next way:

- Systems assign that player as winner between players that died before than him/her and loser front players that are alive until that moment. So then, player will be marked as defeat in the previous table.
- With help of the elo function, system define $R_A = player's\ elo$; $R_B = media's\ elo$; $E_A =$ sum of the expected scores against each play (calculate by $\frac{1}{1+10^{\frac{R_B-R_A}{400}}}$, but R_B correspond to each player in the match)
- Once done that calculate, proceeds to add one by each player that dies before than him/her, this would be S_A result.
- Next, the new player's elo is defined by the next formula: $R'_A = R_A + K(S_A - E_A)$

K factor is defined by the next way:

- K=32 to player which has less than 2100 elo points
- K=24 to player which has between 2100 and 2400 elo points
- K=16 to player which has more than 2400 elo points

2) Stats implementation:

Each player has a history of his games where the position reached, the resources he obtained, the time he spent in the game and the deaths he made is recorded.

To compare players, the system proceeds to evaluate the history of both players, statistics by statistics and a list of advantage and disadvantage is made. The player which has the most profit points will be marked as the best.

To compare players' latency, each player will have an attribute with their ping that means their connection status. Moreover, the system proceeds to calculate whether both players have the same ping o almost equals. Based in a difference threshold, the system compares both player, whether difference is above that threshold, the player which has less ping than other will be marked as in better connexon status. That difference threshold will be defined according the situation. This process is called latency compare.

So then, to compare players, the system, based on results of ranked compare and latency compare define each player as in same or equals conditions whether: both players have the same or almost equals abilities and connexon status. If some player has better possibilities to win and a better connexon status, compare players will notice that both players are not in same conditions. In this step, the system will provide rank compare and latency compare a threshold.

Players that want to play are enqueueing in a priority queue, where priority will be the searching time for a match.

Keeping match is a list where players enquired keep for list filling. The system will have a limit of keeping match list. So then, when some of them will filled, it will be sending to start a match.

Each list has with an interval that define its required latency and ability to add some player. That interval should be given by the first player that was added.

When a player wants to be add in a keeping match list, the system proceeds to compare him/her condition with the first player added. Whether player that wants to be add is considered as same or almost equals than the first player so will be add. In case that player will be rejected in each list, but system has a free list, player will be the first to be added in that free list.

If a player does not be added in a long time, the system determinate which queue has the same conditions that the player and allow that player be added in that list. In the previous case, the system creates a "exception" enlarging the list's interval to compare.

2) Stats implementation variant 1:

Each player has a history of his games where the position reached, the resources he obtained, the time he spent in the game and the deaths he made is recorded. Moreover, the reference to the player who killed him is saved.

To compare players, the system proceeds to evaluate the history of both players, statistics by statistics and a list of advantage and disadvantage is made. Moreover, player's opponent are evaluate too with the other player, thus, whether a player face an opponent much better than him/her and much better than actual second player, this situation provide to the first player with more profit points. But, whether, both players faced with a similar player and be killed, it provides both with some profit points. Otherwise, a player faced a weak opponent that the other player, it would decrease profit points to the player who face the weak player. The player which has the most profit points will be marked as the best.

To compare players' latency, each player will have an attribute with their ping that means their connection status. Moreover, the system proceeds to calculate whether both players have the same ping o almost equals. Based in a difference threshold, the system compares both player, whether difference is above that threshold, the player which has less ping than other will be marked as in better connexon status. That difference threshold will be defined according the situation. This process is called latency compare.

So then, to compare players, the system, based on results of ranked compare and latency compare define each player as in same or equals conditions whether: both players have the same or almost equals abilities and connexon status. If some player has better possibilities to win and a better connexon status, compare players will notice that both players are not in same conditions. In this step, the system will provide rank compare and latency compare a threshold.

Players that want to play are enqueueing in a priority queue, where priority will be the searching time for a match.

Keeping match is a list where players enquired keep for list filling. The system will have a limit of keeping match list. So then, when some of them will filled, it will be sending to start a match.

Each list has with an interval that define its required latency and ability to add some player. That interval should be given by the first player that was added.

When a player wants to be add in a keeping match list, the system proceeds to compare him/her condition with the first player added. Whether player that wants to be add is considered as same or almost equals than the first player so will be add. In case that player will be rejected in each list, but system has a free list, player will be the first to be added in that free list.

If a player does not be added in a long time, the system determinate which queue has the same conditions that the player and allow that player be added in that list. In the previous case, the system creates a "exception" enlarging the list's interval to compare.

2) Stats implementation variant 2:

Each player counts on the accuracy of his shots and a follow-up of the average of obtaining resources, weapons and deaths of the last 20 games

To compare players, the system proceeds to calculate the statistics of each player and then compares them, whether they have the same or minimum difference in their statistics, they will be considered rivals in similar conditions. Otherwise, the player with the highest score is marked as better.

To compare players' latency, each player will have an attribute with their ping that means their connection status. Moreover, the system proceeds to calculate whether both players have the same ping o almost equals. Based in a difference threshold, the system compares both player, whether difference is above that threshold, the player which has less ping than other will be marked as in better connexon status. That difference threshold will be defined according the situation. This process is called latency compare.

So then, to compare players, the system, based on results of ranked compare and latency compare define each player as in same or equals conditions whether: both players have the same or almost equals abilities and connexon status. If some player has better possibilities to win and a better connexon status, compare players will notice that both players are not in same conditions. In this step, the system will provide rank compare and latency compare a threshold.

Players that want to play are enqueueing in a priority queue, where priority will be the searching time for a match.

Keeping match is a list where players enquired keep for list filling. The system will have a limit of keeping match list. So then, when some of them will filled, it will be sending to start a match.

Each list has with an interval that define its required latency and ability to add some player. That interval should be given by the first player that was added.

When a player wants to be add in a keeping match list, the system proceeds to compare him/her condition with the first player added. Whether player that wants to be add is considered as same or almost equals than the first player so will be add. In case that player will be rejected in each list, but system has a free list, player will be the first to be added in that free list.

If a player does not be added in a long time, the system determinate which queue has the same conditions that the player and allow that player be added in that list. In the previous case, the system creates a "exception" enlarging the list's interval to compare.

Webography:

https://www.dofus.com/es/mmorpg/actualidad/devblog/tickets/906138-kolossium-leagues

https://boards.las.leagueoflegends.com/es/c/guias-y-consejos/67N9aKAz-como-funciona-el-sistema-de-ligas-mmr-lp-y-rangos

https://en.wikipedia.org/wiki/Elo_rating_system

https://www.techopedia.com/definition/2228/latency

https://en.wikipedia.org/wiki/TrueSkill

https://github.com/gesundkrank/JSkills

https://en.wikipedia.org/wiki/Lag#Ping

https://en.wikipedia.org/wiki/Ranking

https://support.ubi.com/es-MX/Faqs/000024743/How-Does-Rank-Work-in-R6-Siege