GameGraph Technology and Tokenomics Description

This document is intended for game industry professionals, CTO, analysts and investors who want to understand in detail how GameGraph Protocol helps related parties achieve their goals.

The purpose of the document is to provide the high-level overview of the project architecture.

Scope

The scope of the technical documentation is limited to a brief and general description of technologies GameGraph already contains in its inventory or in plans to develop and integrate. The document gives information about the architecture of GameGraph Protocol, its main components and functionality. It does not contain detailed technical descriptions or source code.

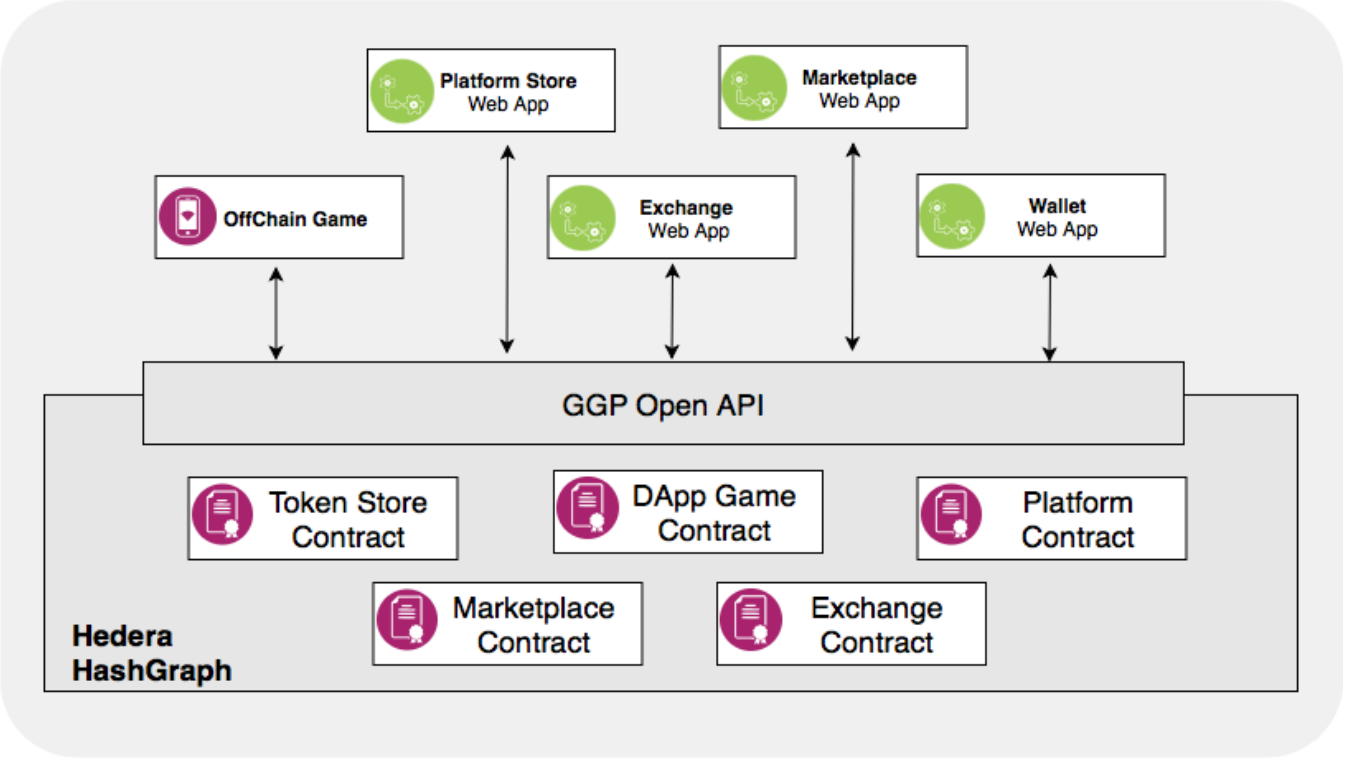
Overview

GameGraph Protocol uses DLT Hedera Hashgraph to store player tokens and data from connected games, convert tokens from game to game, organise single payment service, and comply with common data standards for the related games.

Based on GameGraph Protocol, the Platform is built - a set of technologies, tools and services that allow third-party developers quickly and efficiently create and connect their games or services to Hedera.

1. Implementation

1.1 High Level Platform Architecture



The platform includes the following components:

1. GGP_Platform: Game platform contract
2. GGP_TokenStore: Game token store contract
3. GGP_Exchange: Token conversion contract
4. GGP_Game: Connected game contract
5. GGP_Vote: Voting contract about connecting game to the Protocol
6. GGP_Market: Marketplace contract
7. Exchange service: decentralized token exchange service that operates by AGP_Exchange
8. GGP Open API - open source service for offline games connection to Hedera Hashgraph
9. Marketplace service - decentralized service for marketplace organising based on GGP_Exchange contract

1.2 Smart Contracts

At the heart of the GGP platform, there are several smart contracts, which interact with each other and allow the platform to work in the decentralized way.

1.2.1 GGP_Platform

This contract is the core of the entire system and allows to register games, create a player token and game tokens on the Platform. The tokens contract is based on the ERC721 standard.
The contract is the entry point for accepting player funds redirecting them to the specified game, fixing the fact of payment and the reason for the payoff. It is also the point of exchange of requests for promo materials and tokens within the framework of player tokens transfer system from game to game, without loss of progress.

Data fields and contract methods examples:

playerCreate	It creates a new AGP_Player token during a player Platform registration. It's attached to a specific wallet number and can be transferred from one to another user by the ERC721 standard.
playerCloak	It erases the personal data of the token holder if it fills in.
gameRegister	It connects game contract AGP_Game to the platform. The contract is written according to the AGP standard and contains itself all the methods required to integrate a game into the platform. Registration is carried out by contract decision AGP_Vote.
gameUnregister	It disconnects games from the platform. Disconnection is performed using AGP_Vote contract decision.
gameList	It displays the list of games connected to the platform contract.
playerData	It outputs player data. - Number….
tokenStore	It outputs the address of the contract containing all the platform game tokens.
gameTokenCreate	Attaches game token AGP_Persons to AGP_Player token.
gameTokenBurn	Burns game token AGP_Persons
investorTokens	It gives the list of investment tokens on gamer account AGP_Player.
Pay	It accepts payments from a player AGP_Player to pay for a game service AGP_Game for argument Reason. It's the only method of platform payments to transfer money to a specified game's account. It also charges a fee at the rate set by the platform creator. It takes into account the presence of investment tokens on the player's account and automatically reduces the payment by a discount.
Exchange	Method for exchanging Game A tokens to game tokens B. It calls the corresponding contract method AGP_TokenStore.
getPromo	The platform issues a link to an advertising banner of a random connected game.

1.2.2 GGP_TokenStore

The contract holds all tokens of all games on the platform. It registers new tokens checking them for compliance with platform rules. It converts a token of one game into a token of another one by player request.

Each game stores own tokens independently. Only references to arrays in the game contract get stored in GGP_TokenStore.

Data fields and contract methods examples:

tokens	Array type: GameId, tokenId, OwnerAddress, GGP_Game address
registerToken	When a token created, a game calls this method to verify that the token matches internal platform rules and registration in the shared store.
burnToken	Burns game token
Exchange	It calls GGP_Exchange contract for final conversion of one game token into another game token.

1.2.3 GGP_Exchange

When connected to the platform, a new game indicates weighting parameters in the contract fields - GGP_Exchange template, from which the GGP_Persons tokens should be inherited. These weighting parameters are used to match the attributes of characters in different games. If there is no direct analogy, the data is written in the field GGP_Persons.rawData.
At the beginning of the exchange process, a player receives information about what his tokens will turn into as a part of a new game and decides on the transition himself.

Data fields and contract methods examples:

ExchangeReview	The method is used for gamers to estimate results of future token conversion.
ExchangeDo	It converts tokens

When converting game A token into game B one, and immediately back to game A, we get token identical to the original one. Players pay only for the conversion process.

1.2.4 GGP_Game

The contract implements the logic of the connected game and must comply with the platform rules.

Data fields and contract methods examples:

burn	Contract can be deleted
createToken	method of new token creation, for token visibility on the platform, GGP_TokenStore.registerToken
receive	method for transferring funds from the AGP_Platform contract
withdraw	Transfers funds from the game contract balance to the game owner wallet
GameURL	Link to game site
GamePromo	Link to game advertising materials

1.2.5. GGP_Vote

The voting contract for connecting games to the platform. When connected to the platform, a developer fills in the required form, indicates the link to the game contract and initiates the approval process.
After checking the application for compliance with technical requirements, the platform community votes in due course for the inclusion of the new game and, if approved, the game becomes available on the platform.

1.3 DApps

1.3.1. Platform Store

The application allows players to interact with the list of the Protocol games. The list of the games and their characteristics are displayed.

1.3.1. Token Conversion

The service allows players to move from game to game, without losing their payments. A player indicates what token in what game he wants to transfer, gets information what the token turned into in the game (GGP_exchange used for this). Pressing Conversion button the player receives a new token, which he can immediately use in the game he specified.

1.3.2. Marketplace

Players trade their tokens here. They can sell both his skills (achievements saved in the GGP_Player token) and individual parts of their game progress and experience (GGP_Persons token) or game items (GGP_Atom).
With each exchange, a developer of a game where the token was initially purchased receives a fee for transmitting the token to another player. This amount is deducted from the amount paid by the new owner and transferred to the wallet of the game.

1.3.3. Crypto wallet with crypto-exchanges integration

We will provide a ready-made solution for integrating crypto payments or use the corresponding solution of Hedera Hashgraph core network as soon as it is available for free access. This solution will allow to enter the mass market and attract to the Protocol broad player masses.

A payment made through the wallet service is transferred to the wallet of the specified game for a given reason.

1.3.4. GGP Open API

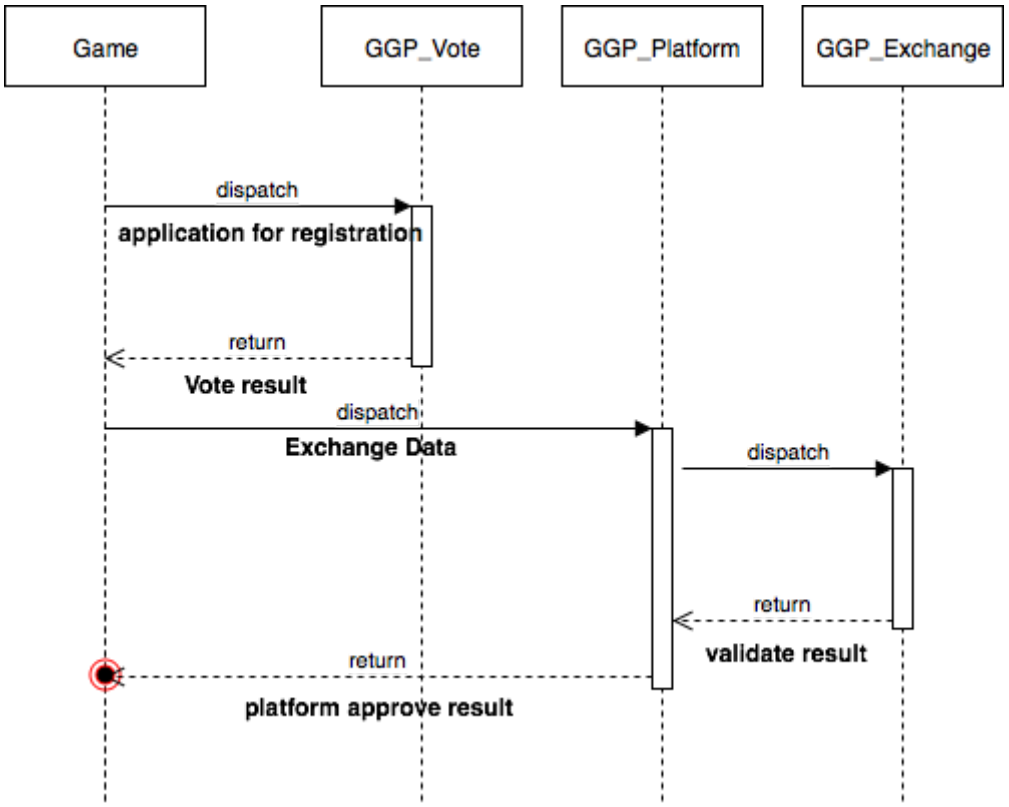
This service provides access to off-line projects to Hedera Hashgraph, allowing interact with DLT through RPC calls to nodes.

Core functions:

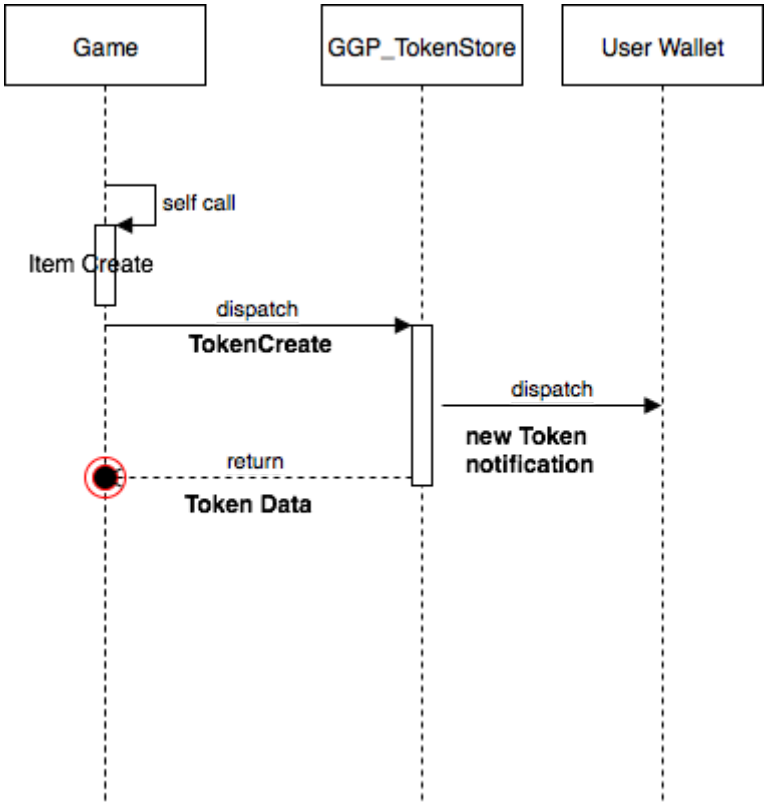
- API for user in-game authentication by wallet ID
- Sending transactions
- View transaction history
- Smart contracts compilation and placement
- Calling smart contract methods
- Subscription to smart contract events

1.4 End-to-End Business Processes

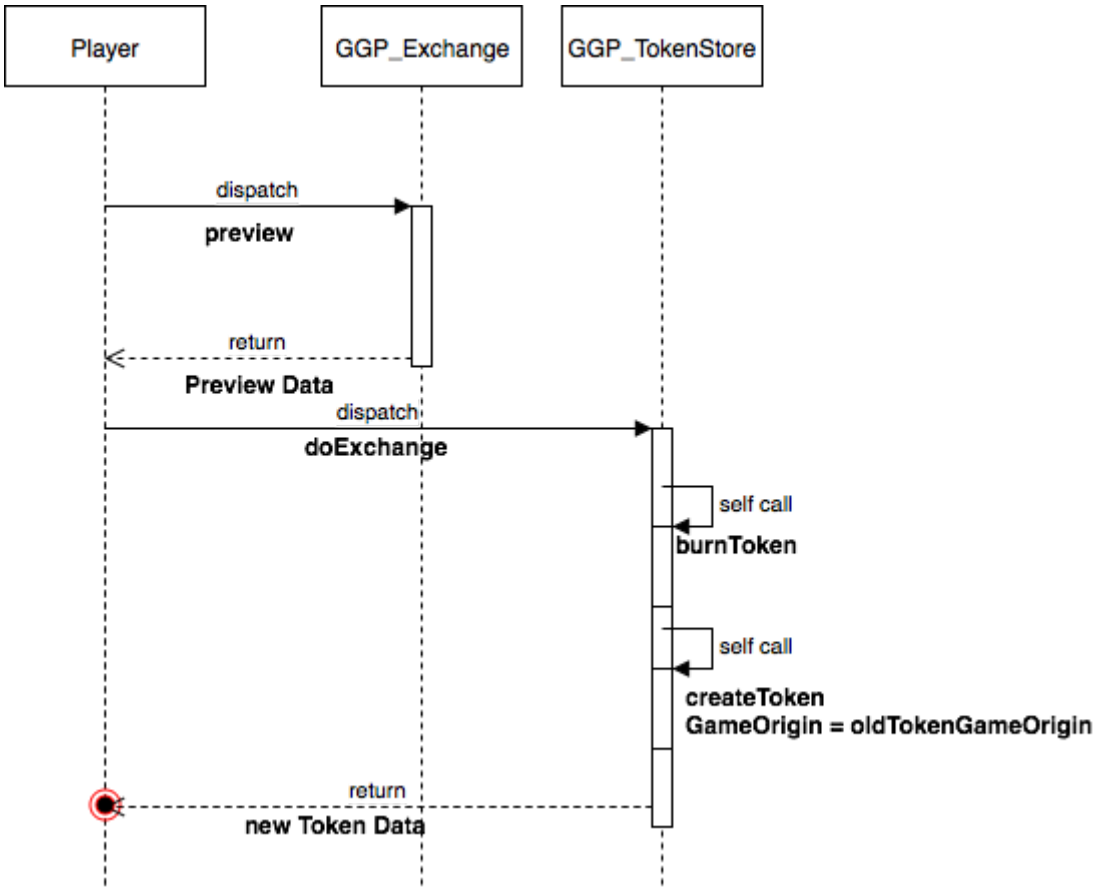
1.4.1. Game Registration Process



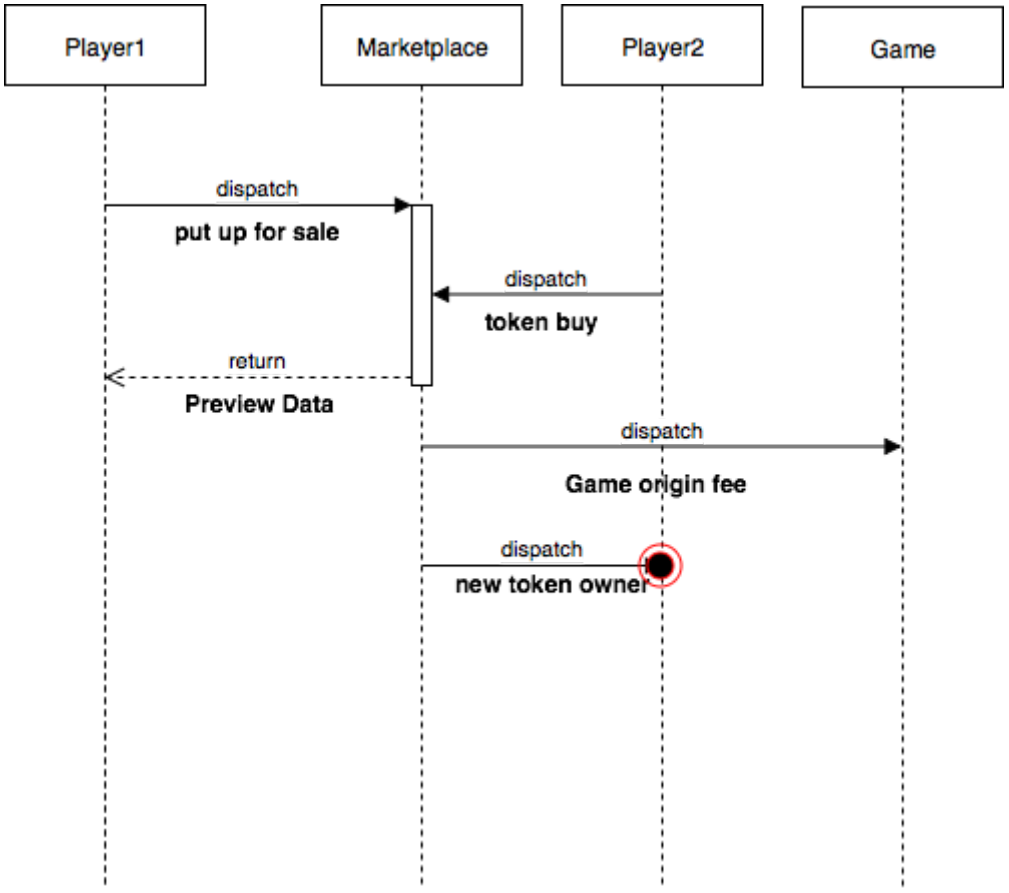
1.4.2. Player- get-an-item Process



2.4.3. Token Exchange Process



2.4.4. Token Sell Process



2. Game Graph Protocol Tokenomics

GameGraph supports two own types of tokens. All payments are made in Hedera tokens.

2.1 IT Token

IT token (Intergame Transferable Token) is an internal GameGraph token that used to transport gamers accounts, digital avatars, virtual items and in-game experience between games and digital universes.

IT token parameters:

1. ERC20 standard compatible;
2. The total token supply is unlimited;
3. It's separable into parts;
4. The token is traded freely;
5. It can be signed up or gifted to another person without limitation.

A gamer register in the GameGraph protocol for the first time and get it for free. He plays any game he likes and move this token between games. Any player may have as much IT tokens as he wants.

2.2. [1]G2 Token

G2 (GameGraph) is a discount and service token. G2 holders get discounts on services made on GamegGraph Protocol. Only G2 token gives its owners the right to open business units on the Protocol (virtual marketplaces, in-game shops, B2B services, etc.)

G2 token parameters:

1. ERC20 standard compatible;
2. The total token supply is 60 000 000;
3. It's available during ICO only;
4. It's separable into parts;
5. The token is traded freely;
6. It can be signed up or gifted to another person without limitation.

The economic model of the Protocol discount token is based on Sweetbridge Foundation research.

G2 is not cancelled (or burned) after use but remains in owner's possession for regular use. Discount token holders activate them in their Hedera wallet to get a discount using GGP services.

GameGraph Protocol calculates a service cost when using G2 according to the formula

$$C(t) = \begin{cases} t \leq \frac{T}{U} & c \cdot (1 - \frac{t \cdot U}{T}) \\ \text{else} & 0 \end{cases} \quad (2.1)$$

[1] https://images.sweetbridge.org/main/WP-Sweetbridge-Discount-Tokens.pdf

where *C(t)* is the fiat cost paid by a user for a given period of time, when activates ***t*** number of G2 tokens in the service's smart contract; *U* is the total number of users who have subscribed to the service at the moment; *T* is the total number of G2 tokens activated on the Protocol.

Example. A product provides its services for $ 10 per month. It has 1000 registered users. In total, 100,000 tokens are activated in the system, used to access various platform services. In this case, one user needs to own 100 G2 to access to the product for free.

2.2 General Formulation of Discount Token Economics

Given the price per unit of service c, we define the cost of services

$$C(t, y; X) = c \cdot y \cdot (1 - f(t, y; X)) \quad (2.2)$$

where *t* is the number of discount tokens activated, y is the quantity or level of service purchased by the user during the license period, and X denotes the global network state or any material characteristics thereof. The function *f(t,y;X)* is the component of the model called the discount function. It is assumed that *f* is formulated in such a way that *C* is always nonnegative, that is, discounts never turn into profits. It is assumed that there is a value *tmax(X),* such that $t \le tmax(X)$, which limits the ability of the user to activate tokens.

2.3 Defining the Discount Function

In the case of GameGraph Protocol, the value y - number of service units purchased – is always 1 as the service is just the right for one user to use software with discount. The discount depends on GGP network utilization and calculated by the following formula:

$$f(t;X) = \frac{t \cdot X_U}{X_T} \quad (2.3)$$

where *t* is a number of tokens activated by a user at the time of the selected service utilization, X_U is a total number of units of services on GGP, X_T is the total number of tokens activated on the Protocol. The discount can reach 100% allowing some users to access the GGP services for free. To do this, a user needs to activate a certain number of tokens calculated by the formula.

Below we explain General criteria for valid discount functions.

Criterion 1. The discount function *f(t,y;X)* must have the property

$$f(0,y;X) = 0 \quad (2.4)$$

for any level of service y, and valid network state X; defining the trivial condition that activating no tokens generates no discount.

Criterion 2. The number of tokens required to eliminate all fees for y units of service is

$$t_{free}(y;X) \quad (2.5)$$

which exists for any y and X, and always satisfies

$$f\,(t_{free}(y; X), y; X) = 1 \quad (2.6)$$

Defining the condition that there is a finite $t_{free}(y;X)$ achieving full discount for any y and *X*.

Criterion 3. The number of tokens required for the 100% discount, $t_{free}(y;X)$ is strictly increasing

$$\frac{\partial}{\partial y} t_{free}(y; X) > 0 \text{ for all } X. \tag{2.7}$$

This ensures that given any state of the network *X*, the amount of tokens required to access the service for free is increasing with the level of use of the service characterized by *y*.

Criterion 4. The maximum number of tokens allowed for activation is bounded by

$$t_{max}(y; X) \leq t_{free}(y; X) \tag{2.8}$$

This bound ensures that hoarding of tokens does not allow incentive manipulation by activating far more tokens than needed for the service actually used.

2.4 Activation Limits and Traction Metrics

Discount token G2 is designed to reflect the overall usage of the network. The choice of traction metrics and the design of the function *f(t,y;X)* will directly influence the token economics once the network is live with active users. Having defined *tfree(y;X),* the simplest version of this solution is linear scaling per unit of service.

$$t_{free}(y; X) = y \cdot t_{free}(1; X) \tag{2.9}$$

where

$$f(t, y; X) = \frac{t}{t_{free}(y; X)} \tag{2.10}$$

and

$$= \frac{t}{y \cdot t_{free}(1; X)} \tag{2.11}$$

where $t_{free}(1;X)$ designates a number of tokens that make a single unit of service free, valid for t in the range $0 \leq t \leq y \cdot t_{free}(1; X)$. The goal here is to design $t_{free}(y;X)$ in such way that t_{free} diminishes as the network gains traction, effectively allowing the token holder to realize the benefits of the growing network utilization directly through an increase in access, either by using more units of service or by sharing or selling the tokens, enabling even more use.

2.5 Effects of the Network Growth

G2 discount token provides its holders access to the Protocol network and allow early users to benefit from the network effects generated as the product gains traction. Here, for every dollar spent by early participants, the discounts realized later in the network's lifetime grow if the network utilization grows.

$$t(y)_{free} = y \cdot \beta \cdot \frac{X_T}{X_U} \tag{2.12}$$

where *y* is a number of ordered units of the service; *β* is a proportionality coefficient of the GameGraph Protocol activation. X_T is the total tokens activated network wide, and X_U is the total number of units of service in use network wide. Users need to take additional steps to activate AGPT and start receiving discounts according to the above formula. This formula confirms that the discount increases as the network grows (parameter U).
The value *tmax = T / U* is called *activation limit*, so users can't activate more than t_{max} tokens. In this case, $t_{max} = t_{free}$, the number of G2 tokens to get access to the service for free.

This formulation provides for reduction in $t_{free}(1;X)$ when X_U increases. This ensures that if you bought tokens early when there were relatively few users, those same tokens would provide the same share of the discounts which, assuming increased network usage, would resolve to a larger absolute discount capacity. That additional capacity could be used by the same user, or transferred to others. The cost of actually using the service remains reliably bounded by c because a user could always choose to buy the service from the network without using discount tokens.

Example. Bob bought 100 AGPT tokens at a price of 0,5$ per G2 at the GGP prototype stage. This number of tokens lets to use the GGP Framework for free. A year later, the amount of G2 tokens needed for free access is 30. The remaining tokens Bob can transfer to another user, sell through the exchange or use to get discounts on other GGP services.

2.6 Deriving Value from Utility

The monetary value of G2 token (token discount value) or the utility value per token activated per unit of time is calculated as follows:

$$U(X) = \frac{C}{\beta} \cdot \frac{Xu}{Xt} \tag{2.13}$$

valid for *t* in the range $0 \le t \le t_{free}(1; X)$, where *C* is a price of one unit of a Protocol service. With increasing of the AGP network usage, the value of the utility function will grow, giving the first platform users the advantage of having tokens at their original cost.

2.7 GGP network commissions

GGP will charge a small commission for AGP services to avoid a discounted negative cash flow (when payments from users are not enough to cover costs of maintaining services). The commission is calculated by the formula:

$$\text{fee} = \frac{1}{\beta} \cdot 100\% \tag{2.14}$$

The service cost includes this commission for all types of users. *(1 / β)* denotes the economic discount factor. This value controls the total cost of GGP services for which the economy can make a discount. β-constant is manually determined once and for all time. The value of *β* controls the size of G2 token activation effect influencing the decline in service prices. A share of expenses is determined by the parameter *Φ = 1 / β,* the recommended value *β = 10 i*s associated with 10% of the cost of all Protocol services compensated by G2 tokens.