Bridge

(White Paper)

Summer 2020

Last update: August 2022

Abstract

Smart contracts will revolutionize businesses by enabling the development of DApps (decentralized applications) and DAOs (decentralized autonomous organizations), which will automate tasks using blockchain technology. The blockchain on which a smart contract is placed and activated, however, cannot access external data because of the underlying consensus algorithms. An innovative technology known as Oracle was suggested as a solution to this issue. When necessary, oracles update the blockchains with information from the actual world. Blockchains cannot be utilized for anything other than tokenization without oracles, however, by integrating oracle technology into a blockchain, smart contracts may be configured to function as a decentralized autonomous organization.

Contents List

Figures & Charts List

1 Introduction

On blockchain networks, smart contracts are a type of program with tamper-proof context and terms that are deployed and executed. It implies that after deployment, no one (including the designers) may change the codes or functionality. Therefore, smart contracts enable parties to come to an agreement and create a new and powerful class of trust without depending on trust in any party or intermediaries, in contrast to traditional paper contracts or digital ones that are programmed on centralized platforms and exposed to alteration, termination, and deletion by a trusted party or a third party. This functionality elevates smart contracts to a more effective tool for creating and carrying out digital contracts.

It is important to highlight that, smart contracts aim to digitalize existing contracts. Consequently, they require real-world data access to complete their duty. Blockchains, on the other hand, avoid connecting to other data sources owing to certain underlying consensus algorithms, so they are unable to reach outside. Because of this connectivity problem, most smart contracts cannot really work, which is a problem for smart contract creators. Without connection, smart contracts and blockchains are merely tools and platforms, respectively, that make it possible to tokenize company shares and other assets; they are not adaptive enough to be utilized as a programming environment.

The Oracle system is a crucial prerequisite for blockchain platforms to be able to handle any type of application and create digital independent organizations by accessing external data. People may add actual facts from the real world to their smart contracts using a tool called an oracle. HTTP/HTTPS API endpoints are the most practical data source for usage as a data feed for inserting information. As previously stated, because of the consensus methods employed by blockchains, they are unable to directly collect such important data. If smart contracts are to take the place of the current manual and digital agreements, it is essential that they are made to be externally aware and able to access off-chain resources.

The following elements constitute the characteristics that the Bridge Oracle System demonstrates:

- The ability to access external data using a range of APIs and parsing helpers including JSON, XML, and HTML.
- The capability to add several data sources, like WolframAlpha, Random, etc.
- The capacity to provide numerous forms of proving the authenticity of the injected data

- Possibility of creating a decentralized oracle service by using technology that exists now.
- The capability to implement specialized Oracle data carriers for certain businesses in order to sell special raw data.
- Acceptance of two different payment methods, including the native coin of the BNB Chain network (Binance Smart Chain) and the project-related token (BRG) with a special discount.

Before proceeding on to the main project, some examples of prospective next-generation smart contracts are given together with their need for external data, which are anticipated to replace their conventional equivalents:

- Usual Contracts: One such contract could be to advertise a video in order to get a particular number of views. The promoter should get paid when the video's views surpass a specified threshold. Access to several views via XPath language scraping is necessary.
- Securities: These include, for instance, bonds and interest rate contracts. Access to APIs that provide market pricing and market reference data, such as interest rates, is necessary.
- Insurance: For instance, if a building's fire extinguisher system is given the capacity to collect fire data, assess the amount of damage, and inject that information into an insurance smart contract, an insurance contract may be able to provide financial recompense. Access to IoT (Internet of Things) data streams is necessary.
- Trade finance: For example, when the transit of goods is completed to a special region, freight is paid using a smart contract. Requirements: Access to GPS data about shipment and alike information in order to fulfil contractual obligations.
    -
    -
    -
- DApp: In general, any type of decentralized application that uses smart contracts as backend codes. Smart contracts may be used to construct any type of application that requires any type of information.
- DAO: In general, any type of decentralized autonomous organization in which any type of information can be used.

The following are the basic reasons for adopting smart contracts in the development of DApps and DAOs:

- No need to host backend code on centralized servers.
- Using blockchain technology to manage contract security.
- Providing the ability to tokenize your platform and establish specialized assets.
- Using peer-to-peer payment mechanisms via native coins and tokens of the same blockchain network.
- Publicly and openly recording transactions that may be used as receipts by customers
- Etc.

It is nearly impossible to create smart contracts without Oracle technology, and without Oracle technology, blockchain's use case is confined to being a tokenization platform. Although large corporations may afford specialized Oracle systems, individuals with restricted money, ability, time, or any other reason are unable to prepare one. However, as you are aware, little companies are more significant all over the world than large businesses and economic corporations. Thus, creating an infrastructure for recruiting small businesses and executing their operations on the BNB Chain blockchain is crucial for the network's progression and success. For example, as the emergence of data centers, server service providers, and ready-to-use content management systems - such as WordPress, Joomla, and others - have led to the easy implementation of small businesses on the internet, creating a public oracle system on the BNB Chain network will allow users to implement their local or small businesses on blockchain, as well as benefit from the BNB Chain network's peer-to-peer payment methods using native coin and tokens. The Bridge Oracle is discussed more in the following sections.

2 Bridge Oracle

2.1 Oracle Structure

As previously said, oracle is a type of technology that allows blockchains to access real-world data. **Figure 1** depicts a simplified schematic representation of the Bridge oracle system:

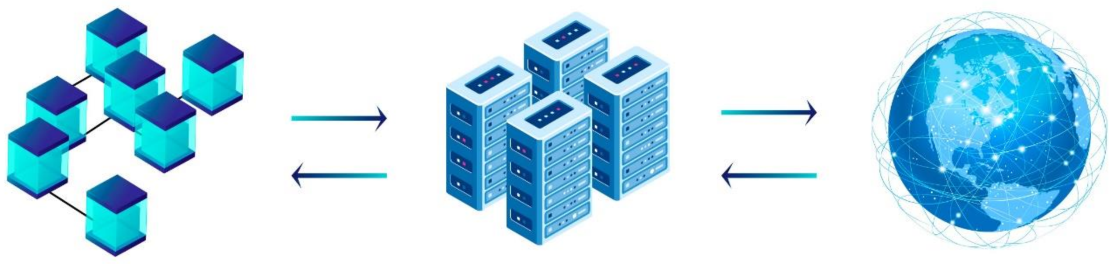

Figure 1. Simple Schematic Structure of Bridge oracle System

2.1.1 Smart Contracts

Bridge oracle consists of 3 various smart contracts that interact with each other. One of the contracts is the Bridge API contract that the client's smart contract should inherit from this contract through which the client's smart contract can connect to Bridge oracle and benefit from provided services. The other contract is Bridge oracle address resolver which redirects the client's requests to the correct service including a public oracle system, a decentralized oracle system, a specific enterprise oracle system, etc. The last contract is the Bridge oracle connector that receives the client's request and processes it in a format of a query and emits specific data which can be read by watching oracle data carriers.

* **Notice**: Note that interactions between Bridge's smart contracts are established in such a way that upgrading the Bridge oracle system to a higher version with more features and without the occurrence of any disruption in previous users' contracts is possible.

2.1.2 Data Carriers

Data Transporters after Bridge oracle connector contract emits information in the blockchain, Bridge oracle data carriers that are observing connector smart contract receive a collection of information from which they recognize blockchain needs of real-world data.

2.2 Oracle Performance

Let's look at how an Oracle system works in general. Considering Figure 1, in the first stage, blockchain emits unique data via logging events that motivate its real-world data requirements. In the second stage, oracle data carriers who are observing such occurrences comprehend blockchain's data demands and attempt to retrieve them from the real world by making requests to various data sources using various techniques such as APIs, downloading files, and asking questions, etc. In the third stage, the targeted resource responds to the request of the oracle data carriers and sends the requested data to them. In the last stage, oracle data carriers insert the final result into the blockchain, to satisfy the blockchain's demand for external information. **Figure 2** shows in detail the interaction of smart contracts of the Bridge Oracle system with each other and with Oracle data carriers. As shown in **Figure 2**, the relevant API contract should be imported into the user's contract depending on the owner's needs in the first phase. There are several API contracts, each with a unique parameter that connects the user's contract to the right connector contract. After importing the relevant API contract into the user's contract, the user's contract must know the address of the appropriate connector contract in order to submit the query in the second phase. So, in order to achieve the address of the appropriate contract, a certain parameter is supplied to the Oracle address resolver contract via the API contract. In the third phase, the Oracle address resolver contract uses a hash table to identify the correct connector contract address. This hash table gets a specific argument that indicates the right connector contract and delivers the connector's address to the user's contract. In the fourth phase, the user's contract transmits the query to the specified connector address, which is controlled by a certain type of oracle data carrier, such as public, decentralized, or enterprise. The connector contract organizes inputs data in a standard manner and logs associated events based on the user's needs and requirements in the fifth phase. In the sixth phase, legitimate oracle data carriers that are monitoring the connection contract get the user's request in an event format. In the seventh phase, valid oracle data carriers issue a request to the data source specified in the user's contract and request the needed data. The required data source responds to data carriers' requests and delivers relevant data in the eighth phase. Oracle data carriers feed users' desired data into their smart contracts in the ninth and last phase of the workflow. To avoid injecting false data from bad and fraud sources, final results are injected into the __callback function of users' smart contracts, which can only be activated by Bridge's valid oracle data carriers.

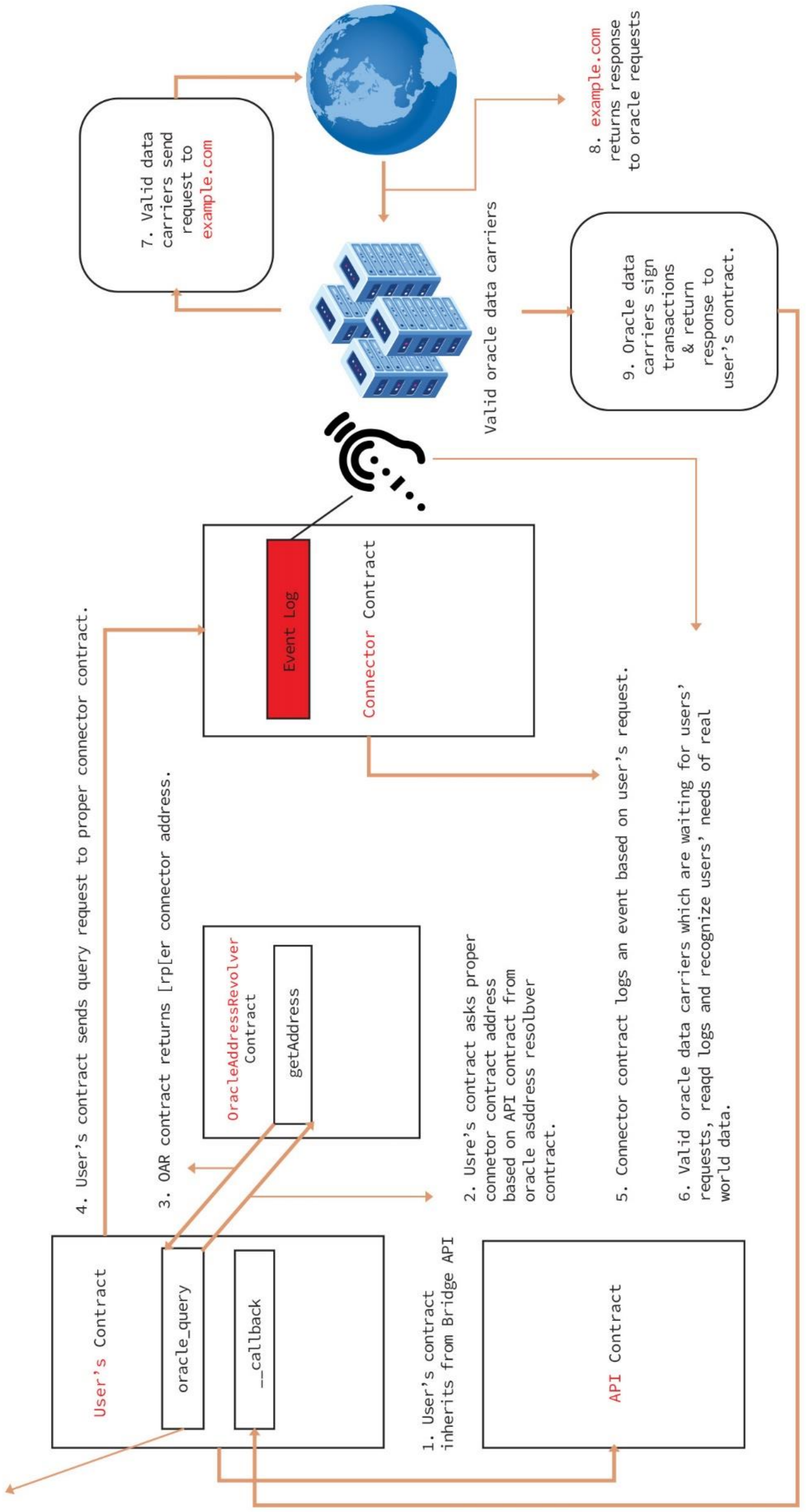
User's Contract
oracle_query
__callback
1. User's contract inherits from Bridge API
API Contract
2. Usre's contract asks proper connetor contract address based on API contract from oracle asddress resolbver contract.
OracleAddressRevolver Contract
getAddress
3. OAR contract returns [rp[er connector address.
4. User's contract sends query request to proper connector contract.
Event Log
Connector Contract
5. Connector contract logs an event based on user's request.
6. Valid oracle data carriers which are waiting for users' requests, reaqd logs and recognize users' needs of real world data.
Valid oracle data carriers
7. Valid data carriers send request to example.com
8. example.com returns response to oracle requests
9. Oracle data carriers sign transactions & return response to user's contract.

Figure 2. Interaction of smart contracts of Bridge oracle system with together and with oracle data carriers

2.3 Oracle API Contracts

According to the requirements of the user, a valid API contract should be imported into the user's contract, as was stated in the previous section. As a result, the Bridge ecosystem has a variety of API contracts, each with unique features. Various API contracts are as follows:

- Public API contract
- Decentralized API contract
- Enterprise API agreements that are specific to each business or organization that voluntarily agrees to sell data for the BRG token It should be emphasized that utilizing any of the aforementioned contracts relies on the user's needs.

2.3.1 Public API Contract

Public API contracts have been created to simply eliminate requirements from external data, as users do not place a high priority on data validity or proof of validity. It goes without saying that many types of proof will be offered in this way to address the issue of data validity proof in the near future.

2.3.2 Decentralized API Contract

Users that need to implement a completely decentralized platform have access to a decentralized API contract. As a result, they also need to use a decentralized oracle system to obtain outside data. The specifics of constructing such an oracle are part of our future goals, therefore future reports on the system.

2.3.3 Enterprise API Contract

Enterprise firms and organizations that control specialized and devoted data that is not shared or published publicly have access to an enterprise API contract. The Bridge system will allow these businesses and organizations to sell this data, and users who require it may import the company's specific API contract. This API contract diverts users' queries to the company's connector, where the company's specialized Oracle data carriers keep an eye on users' contracts, immediately reply to them, and secretly insert specialized data into them. Businesses and organizations may trade their specific data for the BRG token, which is financed by user contracts. Enterprise companies may really profit by selling data to Bridge users. One of Bridge System's business ideas is this one.

2.4 Public Oracle Data Sources

Data sources are different types of reliable references, such as a website or a web API, from which Oracle data carriers request the required data in accordance with users' requests. It should be noted that each data source has a unique use case and functionality, making it crucial to pick the right data source. The Bridge Oracle System is designed to have several essential data sources. The public oracle system for Bridge may handle several data sources, including:

- URL
- Complex URL
- WolframAlpha
- Random
- Nested
- Etc.

2.4.1 URL

This data source is an all-purpose data source that allows users to create any type of data via HTTP/HTTPS request APIs, and the corresponding replies are formatted in one of the following cases:

1. JSON
2. XML
3. HTML

2.4.2 Complex URL

A parameter that is made of a single set of JSON, XML, and HTML parameters may be fetched by a URL data source from HTTP/HTTPS request APIs. There are, however, certain unique situations where one runs into a variety of different JSON, XML, or HTML arguments. One array, for example, included several sets of parameters connected to event matches. Suppose that we need to find out who won a match that had a unique matchID that was included in the array. Such data source is offered to extract data like matchWinner utilizing a deterministic parameter like matchID.

2.4.3 WolframAlpha

This data source transfers users' queries and inquiries to WolframAlpha Company's computational knowledge engine, which can provide the information you're looking for to compute or learn about. The questions are answered by WolframAlpha's artificial intelligence, and the related answers are then sent back to the users' smart contracts.

2.4.4 Random

The contracts of users are supplied with random numbers generated by this data source. Many crucial applications of random numbers include statistical sampling, computer simulation, cryptography, entirely randomized designs, computations in the sciences, etc.

2.4.5 Nested

The usage of multiple queries to the same data source that gives a unique response, or a mix of different data source types is made possible in this scenario for users' smart contracts.

* **Notice 1**: It should be mentioned that only the URL data source is now operational, with more data sources being produced and scheduled to become accessible in a near future.

* **Notice 2**: Off-chain structure has been created based on polymorphism provision, making it simple to implement new data sources. Therefore, users from all over the world can suggest new ones, and the Bridge team will give them in turn based on priority, in order to have more practical data sources in the Bridge Oracle.

* **Notice 3**: Take note that the references to the aforementioned data sources are those that are open to the public, and the information returned in such data sources is available for everyone. In these situations, Bridge Oracle just serves as a gateway to such open, real-world data. However, in order to connect to specialized firms' dedicated connector contracts and oracle data carriers and obtain their dedicated rare data directly from their data carriers in return for BRG tokens, users must import companies' dedicated API contracts.

2.5 Responding Time of Requests

2.5.1 On-Time Queries

Users' contracts that include this option will guarantee that Bridge oracle responds as soon as possible. The requests are processed right instantly, and the outcomes are sent right away, after transactions of users' requests are approved, the necessary information is released in connector contracts, and they are received by oracle data carriers.

2.5.2 Scheduled Queries

By using this option, user contracts can send connector contract inquiries that will have responses due in the future, after a specific amount of time, or at a specific time stamp. By providing an accurate time stamp, users might, for instance, ask the Bridge Oracle to deliver the BNB/USD exchange rate in an hour or even tomorrow at 10:00.

2.5.3 Open Ending Time Queries

Open Ending Time Queries This part has been classified as both sections 2.4.7 and 2.5.3. This is a result of this area being a unique data source with distinct use cases. Additionally, this feature's requests have certain time-dependent functionality. Consider a snooker match, for instance; nobody can precisely anticipate when the game will conclude. Let's say someone wants to inject the match winner so they can do certain operations dependent on the match winner. However, the match hasn't ended yet for a static outcome, and nobody has even predicted the contest's conclusion time. Depending on the user's preference and the capacity of the proposed server, there are three options in such circumstances:

1. Web socket protocol
2. Long polling method
3. Recursive HTTP/HTTPS request method

Users should utilize the first method in cases where using web socket protocol to make a connection to an API link is possible. This method is an optimized approach and receiving responses through this approach into users' contracts is the most affordable method in comparison to the others. In this method, Bridge Oracle data carriers make a connection to the target link using web socket protocol and wait for a deterministic parameter to be responded to. In the example above, Bridge Oracle will wait for the winner parameter of the snooker match to be fulfilled. As soon as the match is finished and Bridge oracle receives the intended response, it terminates the connection and injects the received data into the user's contract. Long polling method is a similar (inefficient) approach which is provided in old servers instead of web socket protocol.

Consider a situation when a user needs to obtain data through a link that meets the same requirements but cannot establish a web socket or long polling connection. In these circumstances, recursive HTTP/HTTPS requests should be used. Users should guess when the event will conclude before applying the procedure. The user should then decide the time step after each, and the Bridge Oracle should then query the most recent result from the intended connection for the desired parameter. These two components are added to the user's contract, which initiates the process and causes Bridge Oracle to make many attempts to achieve the desired outcome. Oracle makes its initial attempt to obtain the result after a predetermined amount of time, as expected by the user. If Oracle succeeds in obtaining the right outcome, it will cut off the connection and return the outcome. If not, the request will be held and retried after the time step. The oracle will keep repeating this process until the desired outcome is obtained. When the Bridge Oracle receives the answer to its most recent query, it will end the task and inject the new information into the user's smart contract.

2.5.4 Recursive Queries

In comparison to the time-dependent functions described before, this section is a specific case that is nothing new; yet, this component has a special use case that should be taken into consideration. This option allows you to continually and repeatedly inject any type of data depending on the desired time step utilizing a variety of approaches. Consider a DEX, for instance, that needs to regularly update the BNB price in its smart contract. Every “n” second, which is referenced to as a time step, the BNB price is injected into the blockchain as part of this smart contract. Recursive queries are the method used to solve such problems. The Bridge document on the bridge.link website may be used to explore how recursive queries are implemented in smart contracts using the Bridge oracle.

2.6 Off-Chain System Architecture

The off-chain system architecture is covered in this section. **Figure 3** schematically depicts the architecture of an off-chain system.

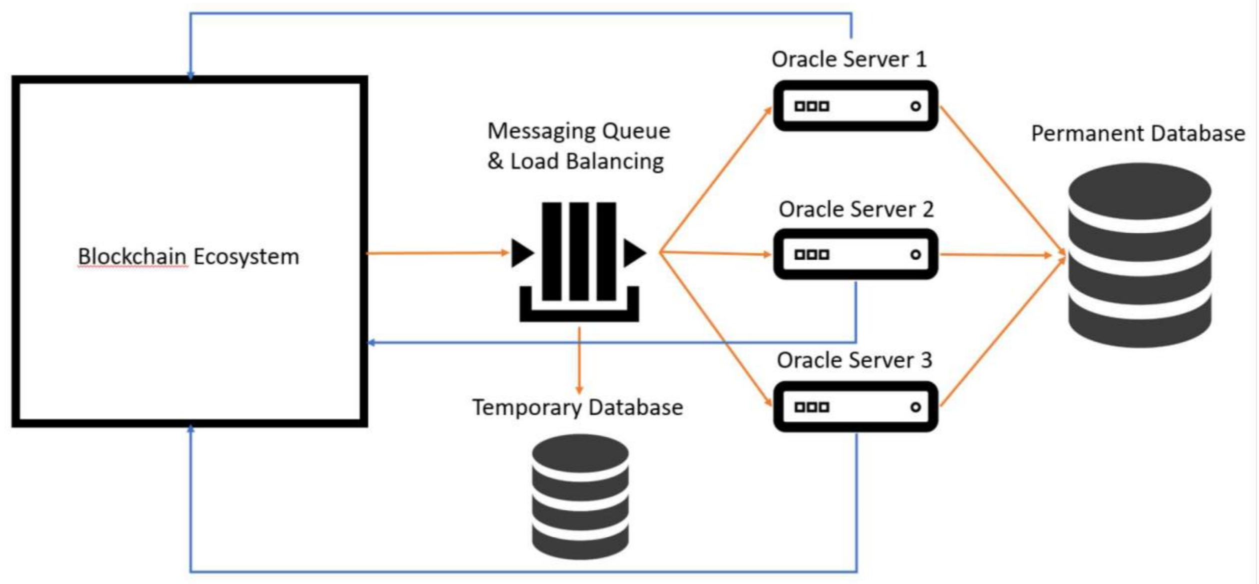

Figure 3. Schematic of Bridge Oracle's off-chain system architecture

2.7 Payment Methods

Methods of Payment users must pay a fee in order to request the injection of data into the Bridge Oracle system. Bridge Oracle offers two different ways for customers to pay for their requests, including:

1. BRG (Bridge token)
2. BNB

According to a certain algorithm, the cost of a request is determined and deducted automatically from the user's contract. BRG or BNB charge the user's contract for access to the Bridge oracle system. Before going into the payment procedure, it should be noted that the cost of the request is determined using BNB. As a consequence, for our pricing system to be able to readily calculate prices based on BRG tokens, the current exchange rate of BRG/BNB must be available at all times. A bot has been created to continually monitor the BRG/BNB exchange rate and compare it to the rate that was previously injected into Bridge Oracle's pricing mechanism in order to solve this problem. If the difference exceeds 1%, this bot adjusts the price, injects the new exchange rate into the blockchain, and bases the cost of subsequent requests on the newly determined exchange rate.

Oracle evaluates the BRG balance of the user's contract in the first step of the bridge. Bridge will automatically charge the user's contract with BRG tokens if there is a sufficient balance according to the cost of the request. In the second stage, Bridge Oracle will examine the BNB balance of the user's contract if there is an insufficient BRG balance. This time, if the user's contract has a sufficient amount of BNB balance, Bridge will charge the user's contract using BNB. The user's request will be denied and no answer won't be considered into the user's smart contract by the Bridge oracle system if there is insufficient BRG and BNB balance in the user's contract.

* Notice: Using the BRG token to charge a user's contract includes a percentage discount compared to using the BNB payment method. Paying using the BRG token is less expensive than paying with BNB to submit a request to the Bridge oracle system.

2.8 Roadmap

It is hard to offer a prioritized roadmap with specific details because of the complexity of the system and use cases; nonetheless, the following general aims are given:

- Including an area for user comments on the official Bridge Oracle website, bridge.link.
- Adding different data sources in order of importance.
- Including several types of verification to demonstrate that users' queries are not modified before being sent to their smart contracts.
- Adding specialized data sources to provide full compatibility with other social media apps, like Instagram, Telegram, etc.
- The following areas are predicted to be covered:
- Sending private post requests using the user's encrypted access token, which should be submitted on the user's behalf.
- Sending requests for public posts that don't depend on the access token's owner, such as the quantity of Instagram posts that have been seen.
- Broad marketing to build an enterprise and specialized oracles for businesses and organizations able to offer specialized and devoted data
- Creating a decentralized oracle system for those who need a fully decentralized smart contract as well as those who wish to participate as nodes in the Bridge Oracle.

3 Conclusion

Smart contracts will revolutionize enterprises and automate organizational processes, but this future is not achievable without real-world data being injected into smart contracts. Therefore, a technological framework is required for the injection of real-world data into smart contracts running on decentralized blockchain networks. This system is an Oracle product.

BNB Chain network claims to be the best platform for producing DApps, and the BNB Chain foundation is constantly marketing the network to attract more developers and grow the network's ecosystem and community. Naturally, the BNB Chain ranks second among blockchains in terms of the number of DApps, with almost 3823 DApps, which is a respectable overall success. However, the Ethereum blockchain, which is ranked first among blockchains, has more than 3000 DApps in place, and this clear difference between the first and second-placed blockchains' DApp counts is surprising. Such a large gap results from a critical deficiency that is felt severely in the BNB Chain. There should be a practical way to close the gap, based on which BNB Chain is able to surpass Ethereum and take the top spot among blockchains. Offering a public oracle system is the solution. Unfortunately, the BNB Chain lacks a public oracle system, which prevents regular users from actively participating in the ecosystem. From their perspective, the BNB Chain only functions as a tokenization ecosystem. Without a public oracle service, it is easy to say that developing a DApp for a regular user is nearly impossible.

As stated at the beginning of the context, little companies are more important across the world than powerful organizations and significant commercial enterprises. For the network to advance and grow, it is essential to provide an environment for attracting small companies and implementing their enterprises in the BNB Chain. Let's talk about a web-based project analogy. in the form of newly developing data centers, server service providers, and ready-to-use content management systems like WordPress, Joomla, etc. That greatly accelerated the adoption of small business implementation on the internet, establishing a public oracle system on the BNB Chain will have a comparable impact and enable regular users to implement their local and small businesses on blockchain in addition to enjoying the peer-to-peer payment methods of the BNB Chain using native coin and tokens. The Bridge Project is a public oracle system that was created to address these problems and offer these services.

4 Appendix

4.1 BRG Tokens

The Bridge Oracle has launched the Bridge Token (BRG) on the BNB Chain. The goal of this token is to provide customers with a payment option that is significantly less expensive than using BNB to pay for Bridge oracle services.

4.2 Token Distribution Policy

The allocation of tokens will be done in accordance with **Chart 1**. The fundamental principle of distribution policy is to prevent centralization and create maximum dispersion. In this phase, an Initial Exchange Offering (IEO) event have taken place on BW.com on September 15th, 2020, where 20% of the tokens were sold and given to participants from all around the world. It should be noted that during the private sale phase, 15% of the tokens were presold. 10% of the tokens are designated for the Bridge Oracle founding team, and 5% are set aside for advisors and consultants. Eventually, 10% of the tokens were allocated to Bridge Oracle's marketing and business development, 10% to software/hardware development, 10% to operations, such as opening new branches and collaborating with other companies, and 20% to a reserved staking pool for operations related to incentives for the decentralized oracle system.

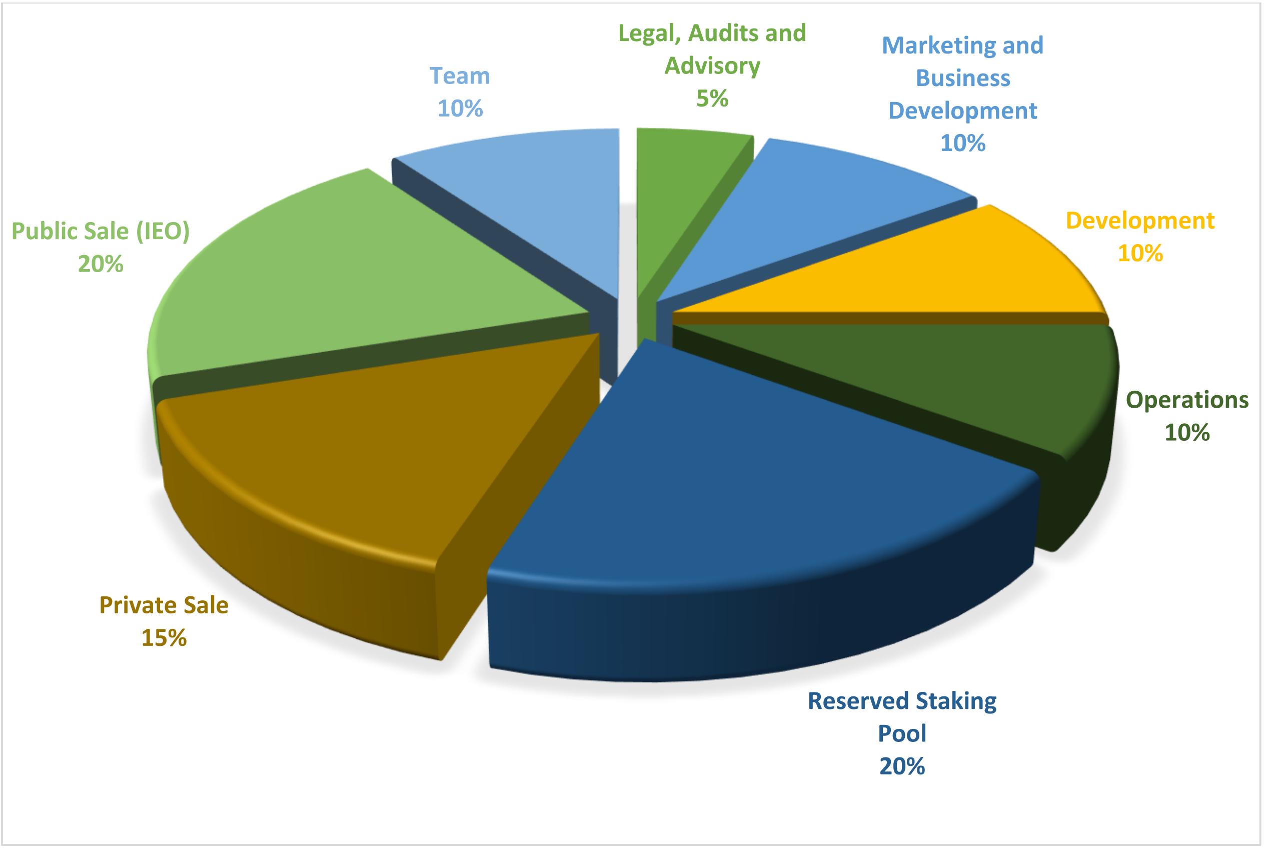

Chart 1. BRG token allocation