Scorpio protocol

Link the protocol layer and application layer to change the hierarchical rationality of the existing architecture Applicability of block chain domain.

Summary

block chain is not a new technology, but a new technology combination. Its key technologies include P2P dynamic networking, shared book based on cryptography, consensus mechanism, intelligent contract and so on.

Most innovations in the history of science and technology are related to productivity, improve efficiency, let people do less work, let machines do more work, the most important subversion brought by block chains is production relations.

The Internet realizes the transmission of information, the block chain realizes the transfer of value, block chain can be regarded as the basic protocol of "value Internet", similar to "information internet" HTTP protocol, the two are the application layer protocol proposed on the TCP/IP protocol.

Block chain is not a omnipotent technology, and in some applications, it does not have obvious technical advantages compared with traditional technology. Therefore, entrepreneurs and investment institutions need to consider the problem of technical applicability.

There may be several platform companies in the underlying technology and protocol layer of the block chain, but most of the

investment opportunities lie in the application layer, the "block chain +" project based on industry applications.

Cryptography ensures the security of data transmission and access, and uses an intelligent contract composed of automated scripting code to program and operate data with a new distributed infrastructure and computing paradigm.

Intermediate protocol layer

The middle protocol layer is composed of consensus layer, incentive layer and contract layer, in which the consensus layer mainly includes all kinds of common understanding algorithms of network nodes, and the incentive layer integrates economic factors into the block chain technology system, mainly including the distribution mechanism and distribution mechanism of economic incentive, and the contract layer mainly includes various kinds of scripts, algorithms and intelligence. Contracts are the basis for block chain programmability.

The characteristics of the disintermediation, the consensus mechanism and the non tampering of the block chain will increase the efficiency of the data transfer, reduce the cost, monitor the real situation of the assets in real time, and ensure the trust of the various organizations of the trade chain to the underlying assets.

The first, first good block chain technology application scenario will

involve many trust entities. We need to have a trust intermediary to cooperate.

Second, there must be a strong cooperative relationship between the subjects, which is the need of business.

Third, the current block chain technology can only be used for low and medium frequency transactions, whether it can meet transaction needs.

Fourth, business models must be complete and sustainable.From the above logic, finance, supply chain management, transportation, energy management, e-government and so on are suitable for the application of block chain integration. In addition, the application of block chain can be divided into two broad categories. One is to use block chain technology to solve some of the problems that have been solved by other technologies today, but block chain technology can better reduce costs or improve efficiency; the other is to solve problems that have not been solved before using block chains. At present, the application of block chain is mainly in the first category.

Will the block chain be a bubble?

At present, there is a misunderstanding that all kinds of movies, music, pictures and big data can run on the block chain. In fact, it is unlikely in the short term. At present, the mainstream public chain technology block is still a few MB size- level capacity. If a lot of data is written

into the block, the size of the block chain will expand to no storage in the short term. If the bitcoin block chain is compared to the DOS system, then the support of intelligent Ethernet contracts is like that windows98.

So, I think we should start preparing for ourselves, because we are going to face a new world, a world where a distributed autonomous system plays an important role.

1.The applicability of the Scorpio scene

As the most important part of the block chain industry chain, the scorpion application service layer includes various application scenarios and cases of block chain, including programmable currency, programmable finance and programmable society.

In the field of finance, in addition to the application of digital money, the block chain has gradually begun to apply in the fields of cross-border payment, supply chain finance, insurance, digital bill, asset securitization, bank credit and so on.

1) Cross-border payment

The pain in this area lies in the long arrival period, high cost and low transaction transparency. Take third party payment company as the center, complete the account, settlement and liquidation in the

payment process, the account cycle is long, for example, the cross-border payment to the account cycle is over three days, and the cost is higher. Taking PayPal as an example, the common cross-border payment transaction rate is $4.4%+0.3, which is brought to the domestic dollar for $35, with the cost of RMB 1.2%.

The characteristics of the disintermediation, transparent and non tampering of the block chain have not been added to the third party payment institutions, which shorten the payment cycle, reduce the cost, and increase the transparency of the transaction. In this field, the Ripple payment system has already begun the experimental application, mainly for the member commercial banks and other financial institutions in the Canadian Alliance to provide the foreign exchange transfer scheme based on the block chain agreement.

2) Digital bill

The pain point in this field is three risk problems. Operation risk: as a result of system centralization, once the central server is out of problem, the whole market is paralyzed; market risk: according to data statistics, in 2018, there are a lot of risk events involving hundreds of millions of money, involving many banks; moral hazard: there are "one ticket selling" and "false commercial bill of exchange" in the market.

The characteristics of the disintermediation of the block chain, the stability of the system, the consensus mechanism and the non tampering, reduce the operational risk, the market risk and the moral hazard in the traditional centralization system.

At present, R3 has jointly developed a commercial paper trading system based on block chain technology, including Goldman, Morgan Datong, Swiss Union Bank, Barclays Bank and other famous international financial institutions, and the functions of negotiable instruments, Bill issuance and bill redemption. A public test was performed. It is completely different from the technical support structure of the existing electronic bill system. This kind of digital bill can further integrate the advantages of the block chain technology on the basis of all the functions and advantages of the current electronic bill, and become a more secure, more intelligent, more convenient form of bill. In china, Zhejiang Merchants Bank has launched the first mobile digital bill application based on block chain technology, and the central bank and the Hang Seng electronics are also testing the block chain digital bill platform.

3) Credit management

The pain points in this field are: lack of data sharing, asymmetric information between credit agencies and users; limited data

acquisition channels in regular market, a large amount of cost in the battle for data sources; the problem of data privacy protection is prominent, and the traditional technical architecture is difficult to meet new requirements.

In the domain of credit acquisition, the block chain has the characteristics of decentralization, trust, time stamp, asymmetric encryption and intelligent contract. It can guarantee the limited and controlled credit data sharing and verification on the basis of effective protection of data privacy. At present, China Ping An is exploring the direction of block chain inquiry, and pioneering companies such as LinkEye and Bubby block chain are also experimenting in this field.

4)Asset Securitization

The pain point of this field is that the true and false of the underlying assets can not be guaranteed; the many participants, the low transparency of the operation and the low transparency of the transaction, cause the risk to be difficult to control. The data pain point is that the transfer efficiency is not high among the participants, the funds clearing and reconciliation between the parties often need a large number of manpower and material resources, and the asset returns can not be used to monitor the real situation of the assets.

Trusting and veracity of trust.

2.Scorpio structure stratified

Unlike other architecture methods, Scorpio has put forward the requirement of consistency from business design to code implementation, and no longer distinguishes between analysis models and implementation models. That is to say, from the structure of the code, we can directly understand the design of the business, naming it properly, and the non programmers can read the code.

However, in the whole process of Scorpio modeling, we focus more on the establishment of the core domain model, and we think that the need to complete the business is a series of operations on the domain model. These operations include changing the state of the core entity, storing domain events, and invoking domain services. On the basis of good domain models, these applications should be easy and enjoyable.

I have experienced many Scorpio modeling workshops, after a round of days after rounds of intense discussion and boring examination, we are pleased to see the field model displayed on the whiteboard with all kinds of color paper stickers, and the sense of achievement is full of everyone's face. At this time of success, people often ask: how can we land on this model? Then the pleasure of everyone's face

disappeared, and the devil's anxiety was changed to detail. But this is our inevitable implementation detail. In Scorpio's primitive methodology, although the metamodel of "Layered Architecture" is given, there is no clear definition of how to stratification.

1)Layered architecture

The implementation of the layered architecture has also undergone several generations of evolution. Until Martin Fowler extracts the layered implementation structure of the following graph, it is gradually recognized by everyone. The Scorpio method is also effectively supplemented, the problem of the model landing is also easier, and the scope of the core domain model is clearly defined: Domain, Service Layer and Repositories.

(summing up the implementation of the hierarchical architecture, note that "Resources" is based on the abstraction of the RESTful architecture, and we can also be understood as a more universal interface to the outside world, Interface. HTTP Client is mainly aimed at the communication protocol of the Internet, and Gateways is actually the logic of assembling information in the process of exchange.

Our core entities (Entity) and value objects (Value Object) should be in the Domain layer, and the defined domain service (Domain Service)

is in Service Layer, and the storage and query logic for both entity and value objects should be in the Repositories layer. It is worth noting that the Entity's properties and behavior are not separated into the two layers of Domain and Service, the so-called anemia model, which proves that such a implementation can cause great maintenance problems. The metamodel definition in the Scorpio tactical modeling should not be changed during the implementation process. As one of the elements in the metamodel, the entity itself should contain the definition of its own behavior.

Based on this model, let's talk about a more specific code structure. Readers who have doubts about this hierarchical architecture can read the original text of Martin. It is interesting to note that the narration of this model is actually in the test article of micro service architecture, which is worth your understanding.

It is clear that when we talk about the code structure, we are aiming at a subdomain (see the strategic design article) after the scorpion modeling (see the strategic design article), which is our clear component-based boundary. Whether further component-based, such as the modularization of the Bounded Context, or the service of a micro service architecture, the core entity is a further possible component-based approach. At the abstract level, the layering architecture of the old horse is applicable to the service oriented

service architecture, so it can be done according to the code structure if further component-based.

The overall code directory structure is as follows:

-Sample/src/

domain

gateways

interface

repositories

services

It can be seen that in fact, we did not create a directory of external storage (Data Mappers/ORM). From the perspective of domain models and applications, both of which are of no concern to us, are enough to validate the input and output of the entire domain model. As for what kind of external storage and external communication mechanism can be "injected". Such isolation is the basis for enabling independent deployment of services, as well as the requirement of testing domain model implementation.

2)Model expression

After establishing the code structure based on the hierarchical architecture, we need to first define our model clearly. As mentioned earlier, this is mainly about the definition of core entities and services

from the process of tactical modeling. We use the C++ header file (.H file) to show the definition of a Scorpio model.

```
namespace domain{
    struct Entity
    {
    int getId();
    protected:
    int id;
    };

struct AggregateRoot: Entity
{
};

struct ValueObject
{
};

struct Provider
{

};
```

```
struct Delivery: ValueObject
{
Delivery(int);
int AfterDays;
};

struct Cargo: AggregateRoot
{
Cargo(Delivery*, int);
~Cargo();
void Delay(int);
private:
Delivery* delivery;
};
}
```

This implementation first explains the metamodel entity Entity and the value object ValueObject. The entity must have an identifier ID. On the basis of the entity, we declare the important element of Scorpio, the aggregate root AggregateRoot. By definition, the aggregate root itself should be an entity, so AggregateRoot inherits

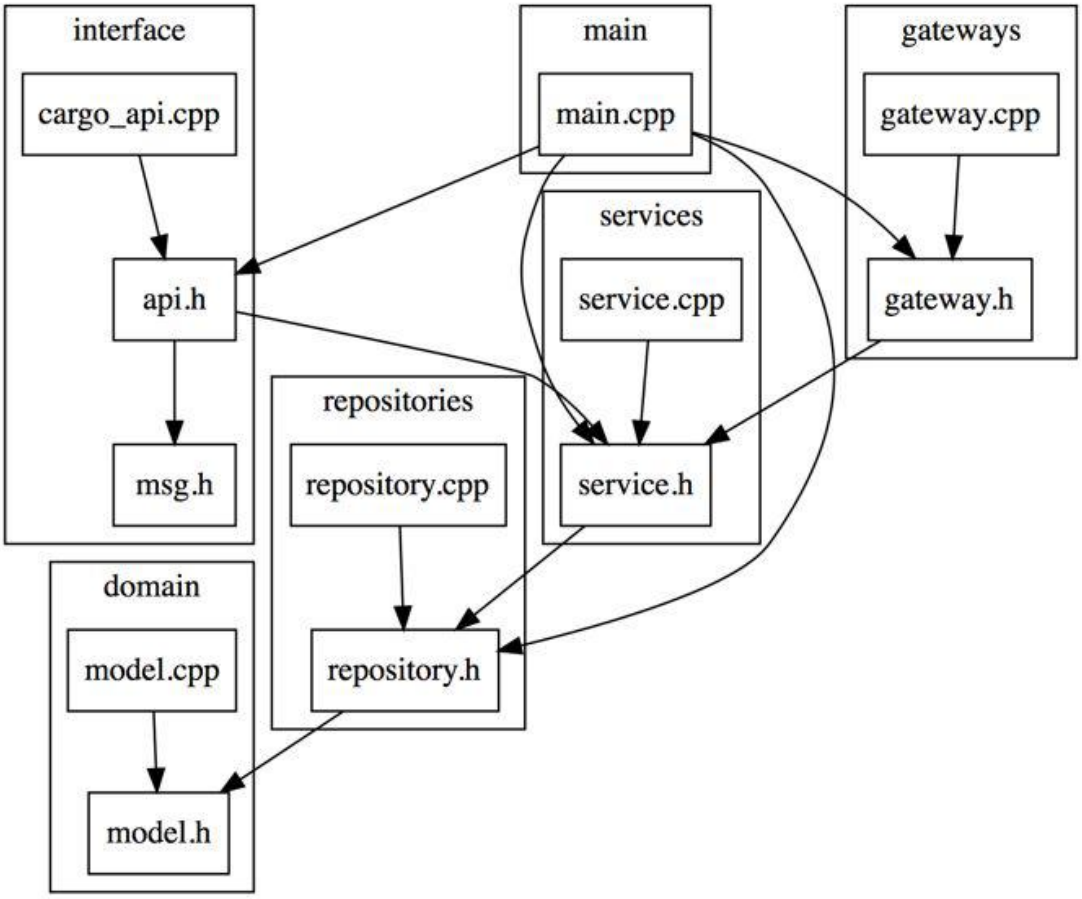
Entity.

In this case, we define an entity Cargo, which is also a aggregate root. Delivery is a value object. Although struct is adopted in order to achieve efficiency, it can be understood in C++ that a class class is defined.

Dependence

The code directory structure does not express the dependencies of all layers in a hierarchical system, for example, the Domain layer should not rely on any other layer. It is very important to maintain the dependence of each layer. Many teams have not been able to establish such engineering discipline in the process of implementation. Finally, the code structure is confused, and the domain model is also broken.

According to the rules of layered architecture, we can see the code structure in the example as shown below.



Domain is not dependent on any other object. Repositories is dependent on Domain and is implemented as follows: model.h is quoted.

```
#include "model.h"
#include <vector>

using namespace domain;

namespace repositories {
struct Repository
{
```

```
};
...
```

Services relies on Domain and Repositories, and is implemented as follows:

model.h and repository.h

```
#include "model.h"
#include "repository.h"

using namespace domain;
using namespace repositories;

namespace services {
struct CargoProvider : Provider {
virtual void Confirm(Cargo* cargo){};
};

struct CargoService {
... ...
};
...
```

In order to maintain a reasonable dependency, dependency injection

(Depedency Injection) is a mode of implementation that needs to be used frequently. As a way of decoupling, it is believed that no one is unfamiliar, and a specific definition is described here.

During test construction, we used a IoC framework (the implementation of dependency injection) to construct a Api and inject the related dependency, such as CargoService, to the Api. This does not destroy the one-way dependence of Interface and Service, and solves the instantiation requirement of Api in the testing process.

```
auto provider = std::make_shared< StubCargoProvider >();

api::Api* createApi() {
ContainerBuilder builder;
builder.registerType< CargoRepository >().singleInstance();
builder.registerInstance(provider).as<CargoProvider>();
builder.registerType< CargoService >().singleInstance();
builder.registerType<api::Api>().singleInstance();

auto container = builder.build();

std::shared_ptr<api::Api> api = container->resolve<api::Api>();
```

```
return api.get();
}
```

For a long time, there are architectural puzzles about this practice, and many senior architects worry that the implementation of the business needs is completely driven by the implementation of an effective technical architecture, and the cost of refactoring can be high at a time. The introduction of Scorpio solves this concern to some extent, and the core domain architecture is identified through early strategic and tactical modeling, which is based on a comprehensive discussion of decision making, taking into account the broader business problems, and more macroscopical than the previously applied business requirements. On the basis of the existing core models, we will also find that the design of test cases is easier to use from the perspective of application, thus reducing the difficulty of testing design.

On pre-design

If the reader does not read the strategic text directly, the reader will certainly put forward concerns about pre design, after all, Scorpio is an architectural approach that our team recognised, and its goal should be to build the responsiveness of the architecture model. And here we are more of a patterned implementation process, like

everything from modeling to coding.

It is worth emphasizing that we are still opposed to the large and complete design (Big-Design-Up-Front, BDUF) of the previous design. But we should recognize the previous analysis and design of the core domain model so that we can help us respond more quickly to the subsequent business changes (that is, the application on the core model). This does not mean that the core domain model will not change in the future, or can not be changed, but that the change frequency of the core part of the unified modeling is much lower than that of the external application. If the core domain model is also changing drastically, we may have to consider whether there has been fundamental change in business, and we need to establish a new model.

We should not forget that our predefined model is also confined to a core problem domain that is decomposed, that is, we do not want to set up all the models in the whole complex business field at one breath. The limitation of this range also limits our pre-designed range to some extent, prompting us to look at the modeling work in an iterative way.

In the end, it is clear that we should have a core team to protect the core domain model, which does not mean that the design and changes of any model must be made by the team of the team. What

we expect is that any modification to the core model can promote wider communication and communication through this core team. The only criterion to test whether a model is landing is whether the team applying the model can reach a consensus on the model itself. On this point, we see a lot of teams continue to practice code review through the way of online and offline communication based on core models, thus playing a real "Guardian" role, making the model itself a common responsibility of the team.

When practicing Scorpio, we still need to follow the core principle of "model is for communication". We hope that the methods and patterns introduced in this article can help you exchange the domain models more easily, and also be a supplement to the strategy and Tactics Design of Scorpio.

3.Scorpio consensus mechanism design

From the practical point of view, the Scorpio system once again discusses the rationality of this consensus mechanism, and naturally derives the concrete implementation mode of consensus mechanism.

1)Why can't reach a consensus on UTXO

We first look at the UTXO based consensus model (in fact, the name is imprecise, in the strict sense that the trading phase consensus

should be called, then we will discuss in detail), which is the common pattern of the typical block chain system: when you want to use or spend a UTXO, each node has to confirm that the UTXO is not Spent, so we need to query the UTXO database to find this object before we confirm the transaction. Verification nodes agree that transactions are established. Transactions (or blocks of exchanges) are recorded on the chain and become "facts".

It is clear that the UTXO consensus model is not applicable to notary based, non broadcast systems: in the case of Scorpio, if a state does not shift between a number of notary, then its current notary can confirm whether he has been used before, without the need of other notary participation.

On the other hand, because the transaction of the system is not broadcast, the other notary in the system knows nothing about the existence of this object, that is, the second part of the article: for the other notary in the system, this object is neither UTXO nor STXO, so other notary can not participate in the transaction process.

In turn, for objects that have changed notary, the UTXO based consensus pattern is also problematic: to implement UTXO based transaction legitimacy checks, it means that all notary knows that the object after the change is still a UTXO when the transaction occurs, that is to say, each notary will record the changed UTXO in the

case.The basis for judging the occurrence of a transaction

In a nutshell, this is indeed feasible, and it is effective for preventing this new UTXO from being double flowers. The reader can verify it by himself. However, because the broadcast is the new UTXO generated by the notary change, and the STXO that has gone through the notary changes is not known by other notary, so it is impossible to avoid the second part of the article, because of the various technical or deliberate frauds, which makes this object again through the way of changing notary. In other notary, a new UTXO is generated to achieve double flowers - the key point is that the UTXO just broadcast is the object of the last notary change, and the double flower attacker is not really the same object by the UTXO generated by the two notary changes and the last change generated UTXO, so no node can pass through U TXO's records judge this double flower.

In short, no matter from the perspective of effectiveness or necessity, it is meaningless to reach a consensus on UTXO in the system of Scorpio mode.

2)STXO based consensus

A consensus is reached for STXO, that is, all notary in the system knows that an object is STXO, and subsequent transactions can be used as a basis for validation. It is clear that all of his historical

information is recorded in a certain notary for objects that have not been changed by notary, so that other notary in the system is still meaningless to know when to become STXO, so we can discuss the consensus process for only notary changes.

Broadcast, when an object has a notary change, the object's current notary should broadcast it to all other notary in the system and tell them, "a certain object has a notary change, so he has become a STXO, and a new UTXO will be generated." Other notary receiving this broadcast should do a series of processing.

Query, in accordance with the rules of the transaction, query their own STXO library to see if the object has been traded before it is already STXO. Note, as we mentioned earlier, in a system like Corda, the query results of a notary to its own STXO database can only reflect an object is not STXO within the scope of the current notary transaction, and the whole system is not STXO, and it depends on the following action, "consensus" - to complete .

The consensus that every notary can return is nothing more than three possibilities: look up / no check / no feedback, and see here, everyone should be able to laugh, that is where the consensus mechanism is in use: not all nodes are bound to feed back the same result (including some nodes because they are not online and no feedback). There must be a suitable mechanism to solve the whole

system's final agreement on "this object is not STXO", which is a typical distributed consensus scenario.

Scorpio does not specify the technical means to achieve consensus, and, according to official claims, Scorpio does not refuse to adopt any effective consensus mechanism, which sounds like a Fabric - supported consensus model. In a word, we should think consensus can be reached. The specific implementation technologies will be discussed in the next section.

It is confirmed that if the consensus is that the object is currently not STXO, it can be traded, and the individual notary in the system can accept the transaction and record the object in its own STXO database to make it a state that has already been traded, thereby eliminating the possibility of the double flower like the future. After that, the notary that releases this change message should also confirm the change transaction and generate the new UTXO according to the result of this consensus. As a result, the notary change transaction was officially completed. Conversely, if the consensus comes to the conclusion that "this object is STXO before", each node denial the transaction through this consensus, and they do not have to do extra action themselves, and the node that releases the change message should not provide a confirmation for the transaction.

3)Several points

At this point, we have finished the basic process of the consensus mechanism for notary changes. Here are a few questions to discuss / summarize the characteristics of the consensus mechanism:

Overall efficiency

It is clear that the Scorpio consensus mechanism is "notary change consensus", that is to say that any transaction that does not involve notary changes is still directly confirmed by both parties through notary and does not need to reach consensus among multiple notary. Only in this way, taking into account the probability of notary change, the performance of Scorpio system can greatly surpass the block chain mode that every transaction needs consensus, which is also the significance of "multi centralization".

consensus guarantee

Since it is a consensus mechanism, it is necessary to meet the conditions required to reach consensus, which is the same as the consensus mechanism of other systems, for example, if the PBFT mode is adopted, more than 2f+1 nodes must be trusted and online; if a mechanism similar to a bitcoin is adopted, a soft bifurcation should be considered. Recovery and so on.

The author here only wants to emphasize that in a system similar to Scorpio, a "strong" consensus mechanism similar to PBFT must be

used, and the "bifurcation" situation can not be allowed. This is because there is no global data structure like "block chain" in the system. Once the bifurcation is allowed, it is difficult to design an effective mechanism to recover, or the recovery mechanism will cause the system to introduce very complex processing processes. Of course, this guarantee should be easier to implement in the alliance chain system, and the existing alliance chain basically adopts strong consensus mechanism.

Malicious notary

Apart from the details, does the new role of Scorpio in notary have any other influence in the consensus mechanism?

I think the most basic thing is to look at the difference between malicious notary and the common malicious nodes in the block chain system, for example, the notary change transaction does not make a consensus on the broadcast requirements, or the consensus process refuses the change, and the notary still completes the change transaction.

This is not the common block chain system, which is the special feature of malicious notary. The solution is also very simple. It only requires the participation of "target" notary in the provision of notary change transactions. There can be a variety of specific participation, and the simplest is that this transaction should let the target notary

participate in the verification so that he can get all the information of the transaction and thus know all the information of the input (STXO) and the output (UTXO).

In the future, when the new UTXO comes to its own trade, it can query the state of STXO in the whole network, see if consensus has been reached, to prevent malicious notary, and still maintain a consensus model for only notary change transactions.

4.The future development potential and trend of Scorpio

Existence is reasonable, block chain technology has such a good advantage, and its future prospects are also very popular. Here I will talk about the future development trend of Scorpio agreement.

1)Integration with the Internet of things

Nowadays, many Internet of things are self-organizing networks of operators and enterprises, and the cost is high. Moreover, the centralized architecture of the Internet of things can not effectively protect users' data security and privacy.

The decentralization of block chains provides the Internet of things with the possibility of solving these problems.

The scorpion protocol uses the unique architecture characteristics to

transmit data through point to point direct interconnection. The whole Internet of things does not need to introduce large data center to carry out data synchronization and management control, including data acquisition, instruction sending and software update can be transmitted through the block chain network. This can greatly reduce the cost.

At the same time, the block chain data itself can not be illegally tampered or lost, so the Internet of things will not cause security leaks and privacy leaks through the data encryption technology of block chain and the P2P network.

2)logistics transportation

For example, if the logistics information is stored in the database, we can check the route and schedule of the logistics at any time because of the transparency of the block chain technology. We can even analyze the past transportation experience and constantly update the best route and schedule, so that the transport efficiency is greatly improved.

3) enterprise financing

At present, many small businesses are faced with financing difficulties. Through the scorpion protocol block chain technology, we can make

the commodity have the characteristics of the asset, and then use the block chain base platform, so that the funds can be effectively and quickly connected to the enterprise, support the innovation and development of the enterprise, and constantly improve the business environment of the small and medium-sized enterprises.

4)traceability anti-counterfeiting

Nowadays, the market is full of fake goods, making it difficult for people to distinguish between authenticity and authenticity. If the Scorpio protocol is introduced, with the function that it can not be tampered with and the data can be traced back, the problem of tracing the source and forgery of commodities will be solved. For example, a block chain technique is used to authenticate diamonds, record the attributes of each diamond, and store the data in the block chain. As long as there are illegal trading activities or fraudulent practices, it will be detected. In addition to diamonds, block chain technology can also trace the source of counterfeit drugs, artworks, collectibles, luxury goods and so on.

5.Scorpio protocol on economy

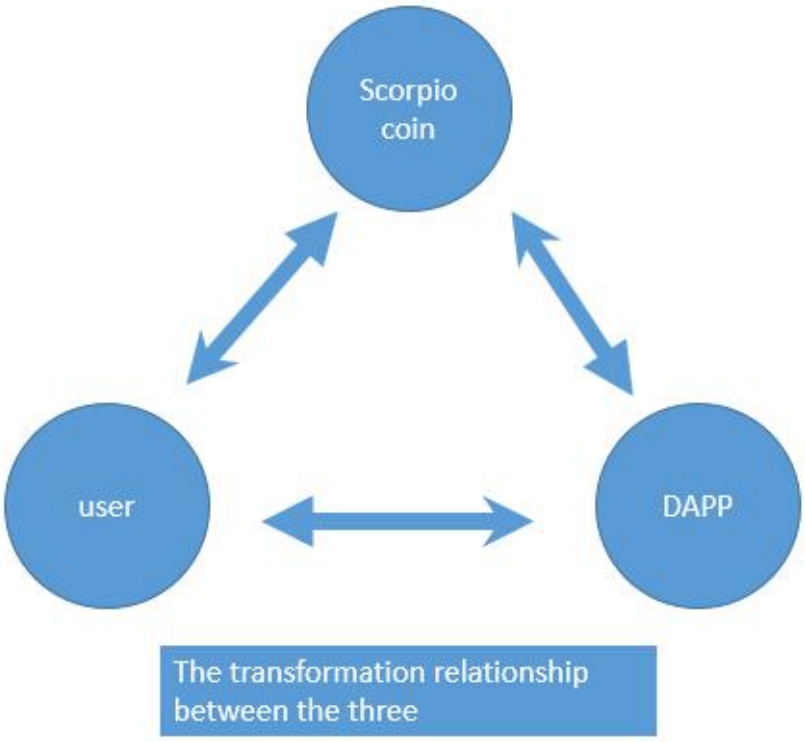
The ScorpioCoin as the underlying token on the Scorpio protocol has

many USES:

1)Dual currency operation

The ScorpiCoin has the dual currency attribute, running the same currency and two token modes on Scorpio protocol. Mode 1: ordinary users hold scorpion COINS, can carry out basic operations such as turning in and out, can hold scorpion COINS to enjoy dividends. Mode 2: DAPP holds the mode. Any application running on Scorpio protocol needs to hold scorpion currency lock as the basic condition for the continuous operation of data in transmission layer. In Scorpio protocol created on DAPP Scorpio currency must have a minimum initial value, is to keep the basic condition of software operation, when DAPP data transmission reaches a certain critical value, DAPP need to increase the cash amount is guaranteeing the normal operation of the program, you can understand as EOS RAM, but Scorpio protocol and EOS and far; In this way, DAPP transmission can be closely connected with Scorpio, and it can also lock in market liquidity and stabilize currency price.

Scorpio's dual-currency model USES the same underlying token, but only for ordinary individual users and DAPP programs, allowing the token to be closely linked to users and programs.

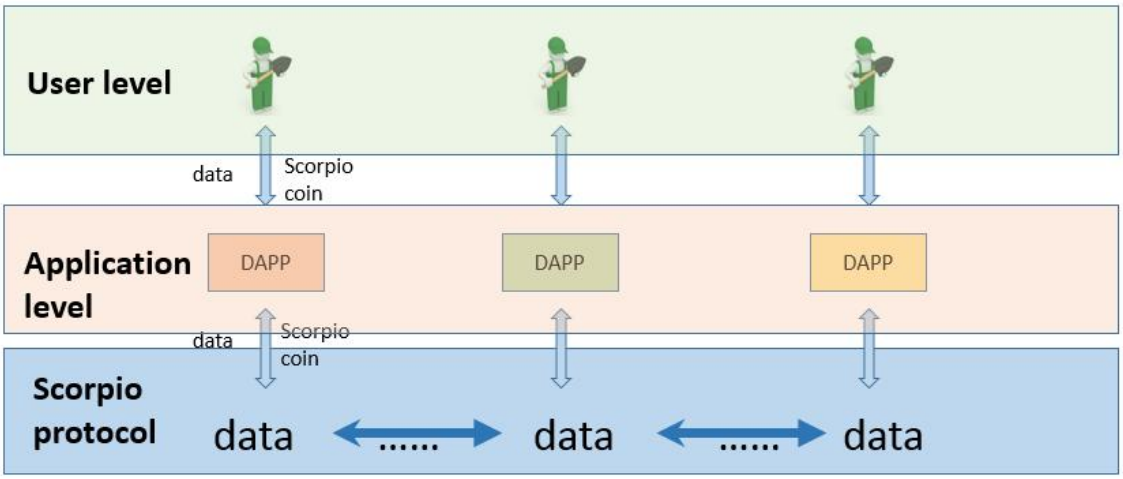


The transformation relationship between the three

2)Data mining

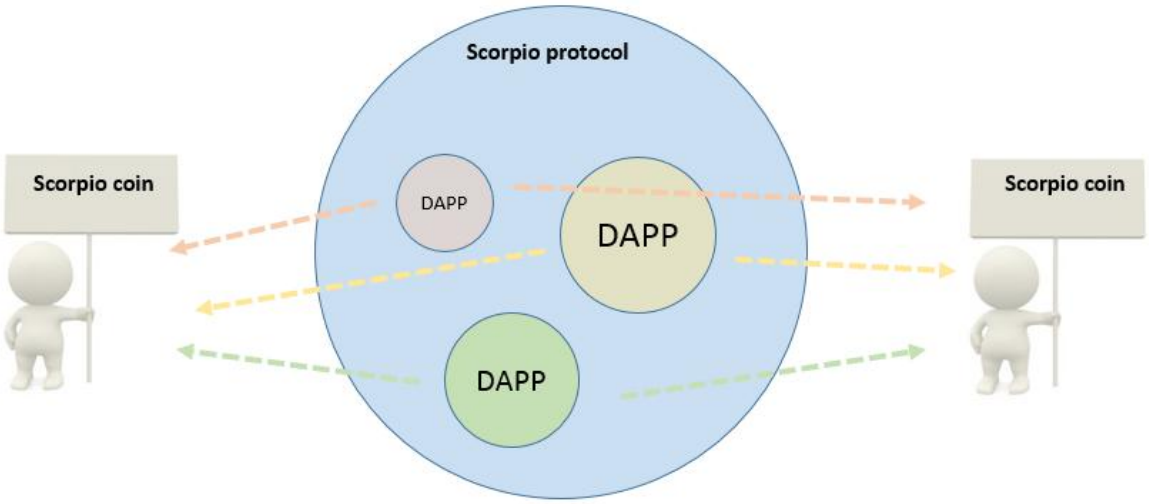
Above we talked about the token basis of Scorpio protocol, and here I need to introduce you to the unique scorpion protocol based on DAPP data transmission mining model. Scorpio protocol as the data transport layer protocol for all upper DAPP transmission chain on data processing, all based on the data transmission will be according to the throughput of the underlying Scorpio DAPP data protocol tokens released, release tokens as DAPP program awarding to DAPP

users. Scorpio protocol automatically assigns tokens to active DAPP contracts or user addresses.



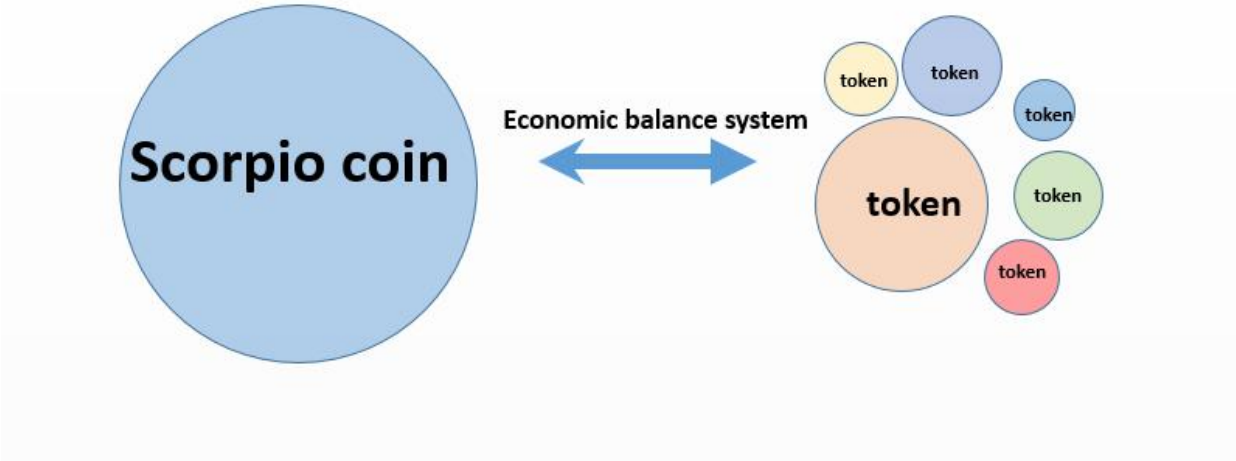
3)Cash dividend

The holders of Scorpio coin can enjoy the right to send tokens based on the DAPP on Scorpio protocol. Who is in Scorpio DAPP preliminary protocol need hold Scorpio currency, or based on Scorpio protocol development own scrip, if choosing in development based on the protocol tokens, Scorpio, Scorpio protocol need to targeted dropped 2% total tokens to Scorpio coin holder, as holding a Scorpio currency benefits of share out bonus. Later on, the more dapps on the Scorpio protocol, the more dividends the users who hold Scorpio coin will enjoy, and the greater the value of Scorpio coin will be.



4) ScorpioCoin pricing system

The scorpio currency pricing system based on all the running in Scorpio, scrip value, the protocol on when the initial run Scorpio currency, Scorpio currency pricing will have a fair value, the value is set, when Scorpio on the network with a total of ten DAPP when (that is, the number of tokens in Scorpio protocol reached 10 species), Scorpio will automatically generate a set of pricing mechanism, Scorpio currency prices will be based on the weighted average of the ten tokens, means to protect and encourage everyone in DAPP Scorpio protocol development, at the beginning of the established DAPP, Scorpio will automatically divided based on its value on to Scorpio tokens of the protocol, Stabilize, encourage, and protect founders to create DAPP.



6.Prospects for the application of block chain

The application of block chain is the trend of future development. It can not only become a new technology of financial infrastructure in the future, but also has a good prospect in many fields.

Block chain technology has obvious advantages, but there are also shortcomings and shortcomings. From the present point of view, the application of block chain to achieve real landing, support the actual business, still need to constantly improve itself at the technical level. The scorpion protocol project will continue to improve and self - complete, make the most mature products and practical business

solutions, and provide a reliable, safe and convenient block chain protocol service for the vast block chain projects.