CIBUS - FOOD ECOSYSTEM ON BLOCKCHAIN
TECHNICAL DOCUMENT

CONTENTS:

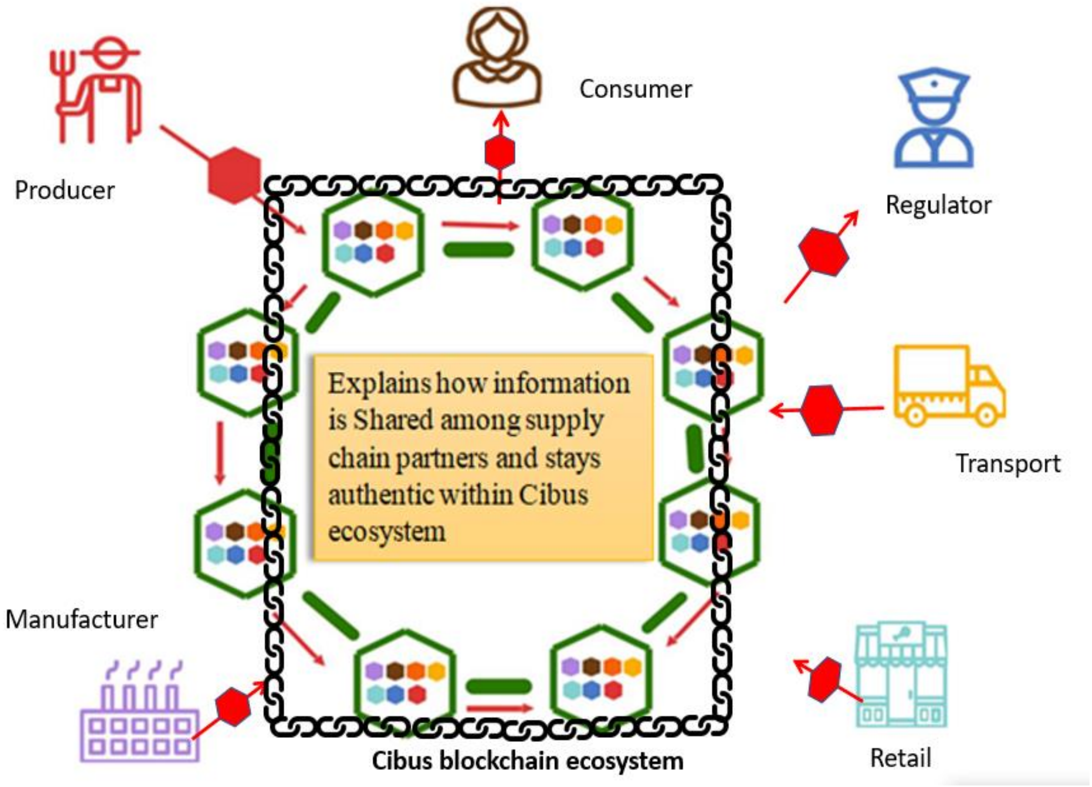
1.INTRODUCTION

1.1 Technical Executive Summary

The CIBUS network is a blockchain-based ecosystem focusing on food and health supplements ensuring the origin, quality and compliance to transparency. The genesis of CIBUS platform is to bring food and dietary supplements manufacturers, producers, sellers or traders under one functional, interactive, and mutually profitable platform based on blockchain technology enabling a transparency via trustworthy data sitting on distributed ledger that can be freely accessed by all interested parties; so that both the business owners and consumers can enjoy best possible transparency, authenticity, and reliability about the food products they are interested in.

Production and distribution records maintained in CIBUS blockchain cannot be falsified. Consumer/Buyers and food producers and regulators could use it to quickly trace food products back to their source, allowing quick removal in case of recall or fraud detection providing high degree of transparency.

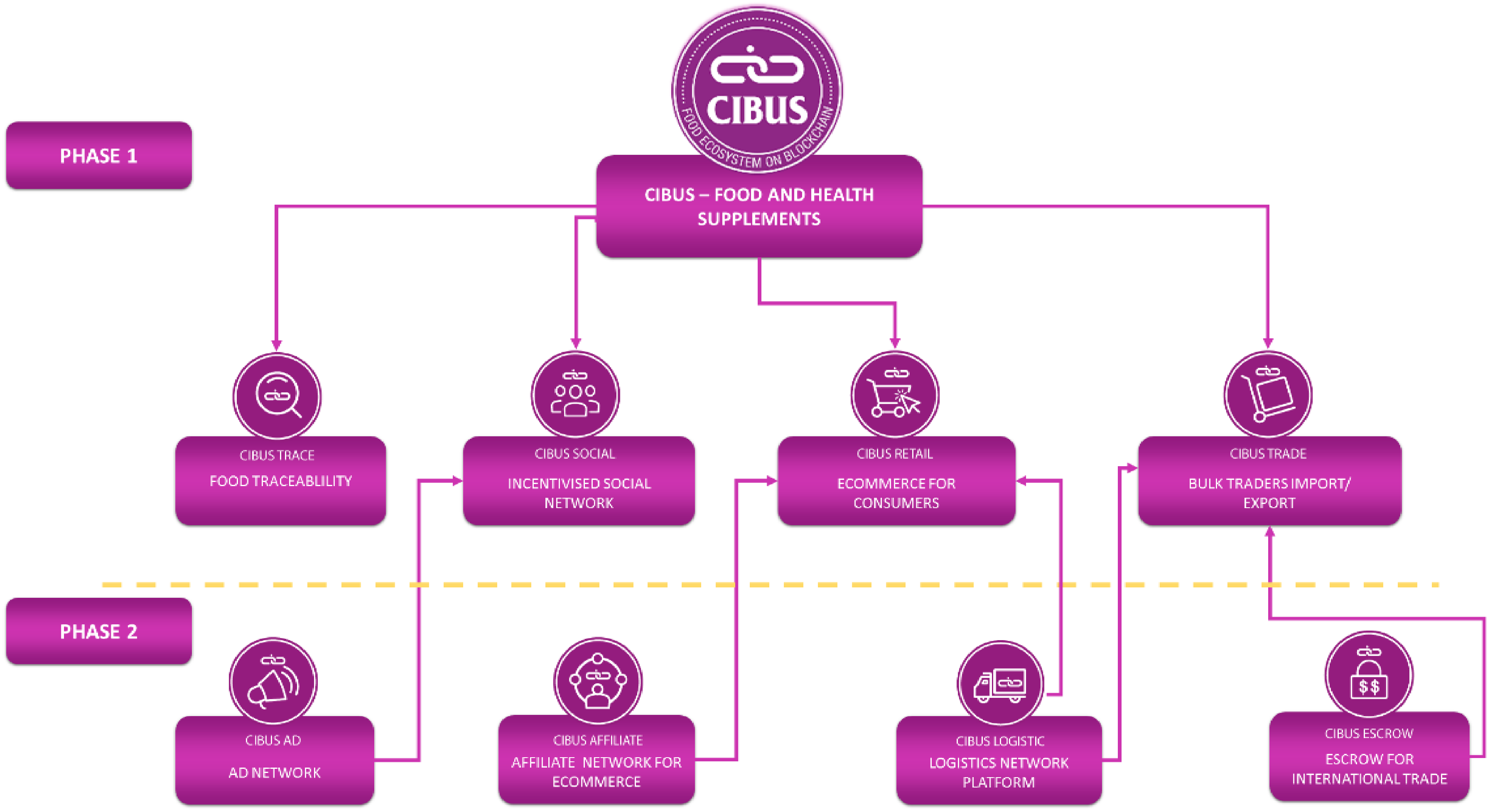
CIBUS Token enables trust less transaction within the platform between all participants. It is the underlying currency for all transactions, mode of payment for transaction fees, escrow services, and advertising and will provide commercial analytics and transactional intelligence.   All transacting parties will be required to buy and pay using CIBUS tokens.


Consumer
Producer
Regulator
Explains how information is Shared among supply chain partners and stays authentic within Cibus ecosystem
Transport
Manufacturer
Retail
Cibus blockchain ecosystem

CIBUS will work as a multi-functional platform that

- Facilitates accessing traceability information by end consumer using CIBUS-Trace functionality: adding complete transparency spanning farm to fork – complete value chain
- Creates interactive global social media platform for food consumers, food manufacturers, trader, food bloggers, food customer's and all other food enthusiasts to enable their scope of interaction with other food manufacturers, traders, experts as well as with a globally expanded health and quality conscious like minded community
- Provides an e-commerce platform that empowers both B2C and B2B trade modality in food and dietary supplements related merchandise using CIBUS-Retail and CIBUS-Trade sub-platforms, respectively
- Facilitates direct targeted advertising by Manufacturers/producers/traders by providing tailored user preference and relevant profile demographic data, which will be the basis of CIBUS-AD

The CIBUS token will be the currency of trade across all the platforms. The aim is to facilitate open and fair trade that reduces uncertainties in supply-chain reliability for the end consumer by developing a holistic, sustainable food centric platform.


CIBUS
FOOD ECOSYSTEM ON BLOCKCHAIN
PHASE 1
CIBUS – FOOD AND HEALTH SUPPLEMENTS
CIBUS TRACE
FOOD TRACEABLILITY
CIBUS SOCIAL
INCENTIVISED SOCIAL NETWORK
CIBUS RETAIL
ECOMMERCE FOR CONSUMERS
CIBUS TRADE
BULK TRADERS IMPORT/ EXPORT
PHASE 2
CIBUS AD
AD NETWORK
CIBUS AFFILIATE
AFFILIATE NETWORK FOR ECOMMERCE
CIBUS LOGISTIC
LOGISTICS NETWORK PLATFORM
CIBUS ESCROW
ESCROW FOR INTERNATIONAL TRADE
$$

The Business model of phase 1 of CIBUS platform will be centered on 4 modules.

1. **CIBUS Trace:**

Traceability is the ability to track any food and dietary supplement through all stages of production, processing and distribution. Traceability also mean that movements can be traced backwards and forward steps at any point in the supply, **which is rightly described as farm to fork traceability function.**

Good traceability systems will help to minimize the production and distribution of unsafe or poor quality products, thereby minimizing the potential for bad publicity, liability, and costly product recalls. The current food labelling system cannot guarantee that the food is authentic, good quality and safe: CIBUS trace facility will put 100% emphasis of offering instant authentication on food safety, reliability, and best quality.

2. **CIBUS Social:**

CIBUS social is an interactive social platform connecting with Food sellers, other users, will allow multi-media uploads, and will soon become a global forum for people with special dietary requirements, user reviews, recipe sharing, user reviews etc.

It will be a food focused community which will be incentivized (with CIBUS tokens) to create a PULL MODEL to encourage Food producers or sellers to share all its product traceability related information on blockchain. It will be mutually beneficial for both the ends - consumers will get to know more about their food products and food businesses will get to know more about the feedback of the consumers directly. Imagine a Facebook for food.

3. **CIBUS Retail**

It is a virtual Business to Consumer (B2C) marketplace based on blockchain technology and cryptocurrency as transaction medium. Consumers will get to buy food products and provide their feedback based on their experience, just like Amazon etc. Buyers can compare food products and dietary supplements based on their traceability data sitting on blockchain coming via CIBUS Trace module. Consumers as well as business operators will have to transact using CIBUS token, which will drive the popularity of CUBUS token over time. Every transaction on CIBUS will attract a micro-fee based on the transaction size.

4. **CIBUS Trade:**

CIBUS Trade is meant for bulk purchase of traders for their trading from another trader or manufacturers. All transactions will be done via CIBUS token; however, just to create the business momentum initially USD transactions will be entertained.

Incorruptible vendor rating system, smart contracts, and most relevant search results based on user profile and past activity, one click purchase advice strategy, escrow services, product tracking facility -will facilitate business to business trade by enabling seamless sales experience. Every transaction on CIBUS will attract a micro-fee based on the transaction size.

In Phase 2, CIBUS platform will be enriched by adding capabilities of the following 4 modules:

5. CIBUS Logistics

It will have a Logistics companies and deals listing for customers to seek quotes and Logistics companies to pick up business. CIBUS may consider at a later date to open its own Logistics centre like Fulfilment by Amazon (FBA) model - to deliver products to its customers from its central warehouse.

6. CIBUS AD:

CIBUS Ads is a direct advertising and marketing platform including data analytics and artificial intelligence. Reduce intermediaries, decentralized advertising platform. Facilitates direct marketing. Manufacturers, traders, or allied business operators who list their products on the website will have the option to purchase analytics-like data on user demographics that can be used to create targeted marketing and promotional campaigns. The advertisement can either be basic or advanced and the fee charged by CIBUS will vary depending on the customization.

7. CIBUS Affiliate

Affiliate marketing is a type of performance-based marketing in which a business rewards affiliates for each visitor or customer brought in by the affiliate's own marketing efforts. CIBUS Affiliate program will provide incentivized reward model for the Marketers for their effort in promoting the sale process of CIBUS Retail and CIBUS Trade modules via promoting their affiliate links and advertising media.

8. CIBUS Escrow

CIBUS escrow is a financial intermediary arrangement where CIBUS will hold and release payments to buyers and sellers involved in a given transaction. It helps in processing transactions more protected and secured by keeping the payment in a secure escrow account, which is only released when all of the terms of an agreement are met as overseen by the CIBUS escrow unit. CIBUS will work as an intermediary between buyers and sellers and all transactions will be facilitated by CIBUS Token. Prominent example will be Alibaba currently providing escrow service between its Sellers and Buyer on its platform.

2.WHY BLOCKCHAIN?

Traceability and lucidity are some of the most vital basics of logistics. Blockchain optimizes business transactions and trading associations with strong and secure business networks on Blockchain—in accordance to scalability and globally.

Blockchain provides a shared ledger that is rationalized and validated in real time with each network contestant. It empowers equal visibility of activities and it can expose any asset at any point in time, regardless who owns it and its latest condition. It is safe, secured, and cannot be tampered by any means so it promotes best transparency.

Transparency is the most challenging aspect of customers that has been increased in recent years. The transparency has the power to evaluate the performance of a supply chain and thereby boost the self-confidence of related players as well as to improve the customer's trust. The strengths in the transparency of the blockchain are in the honesty as each transaction is unique and none of them can be changed or manipulated afterwards. The sovereign trustworthiness cannot be imposed and exercised by a centralized system, since it is not possible for an outsider to evaluate the trustworthiness of the revealed information. Therefore, it is advantageous to use a blockchain technology over a centralized system in terms of trustworthiness.

2.1 Introducing CIBUS ecosystem

CIBUS Token:

Underlying crypto-currency for all transactions on CIBUS platform, which is the only mode of payment for transaction fee, escrow services, advertising and analytics data. All transacting parties on CIBUS Trade and CIBUS Retail need to maintain a minimum CIBUS token balance to initiate their deals on CIBUS.

CIBUS WORLD – GLOBAL FOOD BUSINESS ECOSYSTEM ON CIBUS BLOCKCHAIN

CIBUS transparent global food business ecosystem on blockchain technology can promote following categories of food businesses on global platform:

- Agricultural production
- Industrial manufacturing of processed foods
- Wholesale or retail food distribution

CIBUS blockchain based business ecosystem can offer well suited solutions to the majority of recurring quality and authenticity problems in food business supply chains worldwide. Benefits can be enjoyed in following areas:

- Traceability of ingredients and packaging materials for best food safety
- Brand protection resulting to brand equity establishment
- Consumption of real time data to adhere to regulatory audits
- Monitoring of delivery condition of temperature for sensitive products
- Ensuring compliance to GFSI motivated VACCP (Vulnerability Assessment and Critical Control Points) guidelines.
- Fast recovery of food safety data in case of product recall
- Accurate traceability implanted in supply chain will enable partner producers to deal with problematic products fast.

2.2 CIBUS Architecture

CIBUS Blockchain will contribute in adding transparency which will help global food industry in quick & efficient fraud prevention.

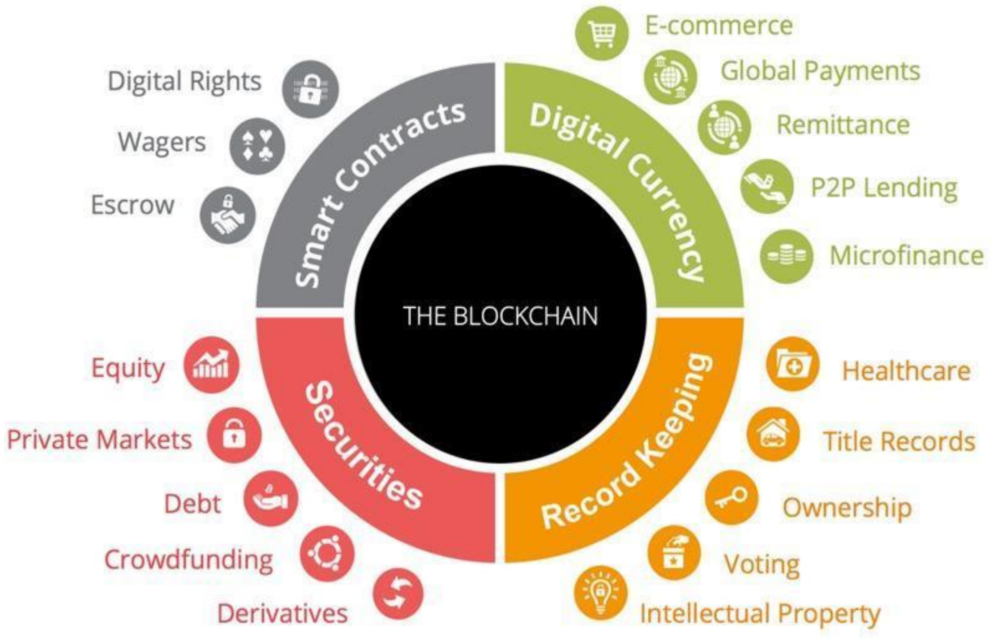


figure: CIBUS Architecture overview

3.CIBUS PHASE 1:

3.1 CIBUS Trace

CIBUS Traceability is conceptualized on the basis of five major aspects. These are safety and quality, economic viability, social, regulatory, and technological. These are considered as main pillars of food traceability idea and all the CIBUS Trace policies spins around them.

The primary objectives of food traceability system is not only to trace, rather it is designed in a way to improve the whole supply chain management through proper implementation of safety measures. This helps to establish an ideal model for raw material supply, production, marketing and consumption of product. Food traceability allows total control on the food supply chain by individual and group identification to achieve the following objectives:

CIBUS Food supply chain Traceability System is based on following six factors:

Major Objectives	**Major Strategies**
● Ensuring food safety through managing risks related to food products and Livestock	● Tracing outbreak in animal ● Identifying sources for hazards ● Ensuring safety alerts ● Removing/withdrawing non-compliant products which is potentially hazardous
● Providing reliable and authenticate information about a product to the users which ensures that they get an authentic product	● Creating fair trade environment ● Removing unfair competition from market ● Safeguarding customers from fraud
● Finally to improve and stabilize the whole supply chain to ensure quality and authenticity	● To identify and restricts non-compliance in the process ● Better management of flow of product from producer to end user

CIBUS will introduce a decentralized ecosystem empowering consumers to save up to 40% on daily food and dietary supplements shopping buying directly from grocery manufacturers. It will be done by:

- Direct interaction between consumers and manufacturers
- Empowering manufacturers to market their goods directly to the consumers with authenticity
- Exerting buyer power when there is no data abuse
- CIBUS will help food manufacturers and traders to present more authenticity to consumers
- CIBUS will cut off food miles: also it will enable consumers to access local food manufacturers, including farmers by implementing "pull" system and it will reduce the problem of inventories and out-of-stocks issues that ultimately will reduce the problem of food waste.

The CIBUS trace will become a reliable food marketplace where consumers can purchase products directly from manufacturers, enjoy low transparent prices for a wide range of high quality products, and save on direct promotions while manufacturers can compete for consumers and interact with them directly by CIBUS Social. All transaction will be done by CIBUS Token and all users have to maintain a CIBUS account/wallet on CIBUS Platform.

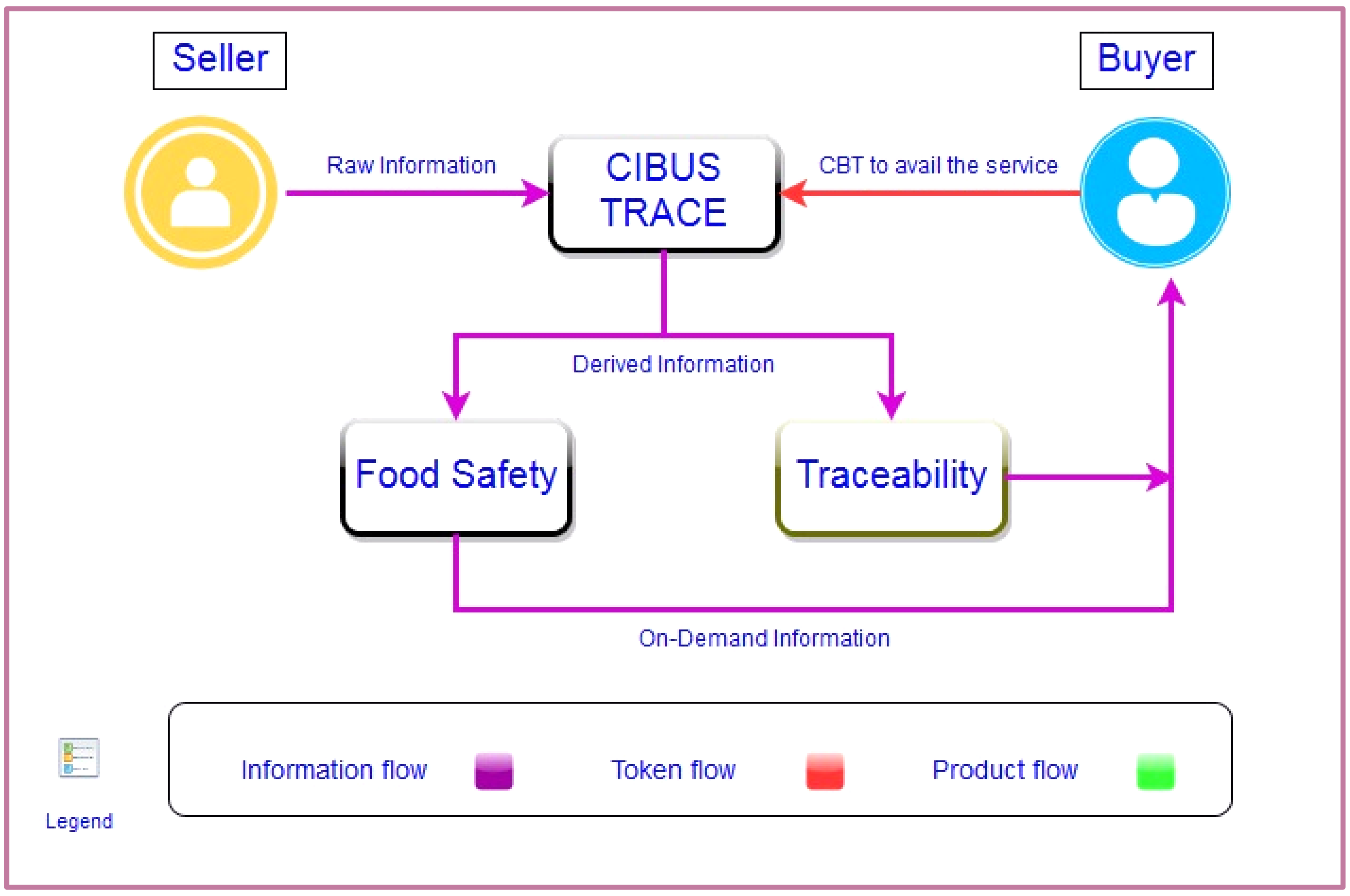


figure: CIBUS Trace

3.2 CIBUS Social

CIBUS social will be a multilingual platform and will be powered by inbuilt automatic translators to help users understand food ingredients in their vernacular language. CIBUS social will be built on pre-fixed features like:

- A social network platform for users to share, discuss and collaborate on food and related ideas: Recipe sharing, multimedia uploads, reviews on food products, AMA sessions with experts, nutrition facts, global forum for people with any kind of special dietary requirements, and crowdfunding campaigns.
- Existing social networks tend to manipulate user news feed to influence public opinion. CIBUS -Social will be designed to ensure that end-user opinion is accessed by food manufacturers and producers.
- The reviews and opinions posted on CIBUS Social will be integrated into the vendor rating mechanism used in CIBUS Retail and CIBUS Trade.
- Voting and polling method enables users to anonymously record their opinions on topics of popular interest.
- Stringent measures to prevent bots from posting or participating in polls.
- CIBUS tokens can be utilized by the users for participating in crowdfunding campaigns.
- Data and media uploaded on CIBUS Social will be stored using blockchain on decentralized servers.
- All uploads will be protected using end to end encryption.
- Users have complete control on who can view their posts and uploads. Additional approval mechanisms to enhance privacy.
- Completely encrypted private messaging options between connected users.
- Connected users can request and send CIBUS tokens to one another privately.
- For social media posts that are in the nature of recipes or nutrition and diet related advices a reliability meter will let other users or experts to rate the extent of accuracy and validity of the post.
- All user posts to be categorized into topics using hashtags and hash tags threads for ease of access.
- CIBUS wallet bonus for existing users who complete successful friend referrals.

- CIBUS wallet bonus for maintaining activity streak over extended period of time.
- Dummy user accounts used for mass disliking or poor ratings will be monitored using statistically predetermined user patterns and such accounts will be blocked.

Benefits of CIBUS Social Interaction:

- Customer reviews are helpful towards future enhancements and improvements of a product.
- Customer independence towards expressing authentic feedback.
- Easy to advertise and attract potential customer in minimal cost.
- Ease to get collaborative information from users.
- Getting meaningful insight to improve business policies and help to enhance better customer retention programs.
- Inbuilt Social Platform to post and review.

Related information posted in other social platforms will also be fetched as per company policy.

Manufacturer/Producer Perspective:

1. Producer can insert product information from an advertisement perspective:

- Product related information.
- Relevant documents.
- Striking features.
- Advantages of this product over existing products.
- Other information.

2. Producer can reply or start/write a social conversion perspective:

- Necessary information.
- Upload relevant documents.

Consumer Perspective:

1. Consumer writes review/blog or took part in a review conversation :

- Necessary information.
- Upload relevant documents.

Social Incentive at CIBUS Social:

➢ Social platform is a goldmine for sharing information, idea, emotion and interaction. The capability of social network including aforementioned activities are obvious but not limited to any pre-defined set. Social platform can leverage the capability to reward the user in different ways.

➢ CIBUS incentive will be distributed after following some protocols of the company in terms of CIBUS Token.

➢ Incentivized category can be but not limited to the following:

- CIBUS Blogger
- CIBUS Reviewer
- CIBUS popular post owner
- CIBUS Active user

➢ CIBUS incentive calculation will depend on following factors:

- Voting system

- Judgment of review quality and honesty
- CIBUS company protocols
- CIBUS company performance and global economy

Problems with general non-incentivized system:

- Existing solutions share a number of problems, most notable of which is false reviews.
- It is believed the reviews listed on general medium are inclined to the negative due to the lack of incentive for the standard end-user.
- In short, end-users who have a neutral or positive experience are far less likely to post a review than an end-user who had a negative experience.

Benefits of social incentive:

- Humans are inherently social beings, and like to be recognized for their expertise and achievements. Recognition via CIBUS Token can be a powerful motivator for social activity.
- Giving people something unique or exclusive in return for sharing can be a powerful motivator — we all want to feel privy to something special by acquiring different badges like CIBUS Blogger etc. As mentioned in 3rd point of above section.
- High quality honest reviews at CIBUS Social platform increases trust with our Decentralized Review system. Authorized Users can review the post and can determine the authenticity via our Distributed Smart Contracts leveraging Blockchain.
- Reviews Given in other linked Social media platforms for the relevant products will also be incentivized if the customer has a CIBUS account.

CIBUS voting system leverage the capability of weighted voting ecosystem:

Vote from different users will be calculated based on the following factors:

- Each user will be awarded to different CIBUS points depending on their activity on the platform.
- A good honest review or an excellent informative blog regarding products or regular active user will be awarded with some CIBUS points. When these users will vote or review a particular artifact that will carry more relevance and importance regarding incentive calculation.
- For an example a certified CIBUS Blogger will have more CIBUS points rather than a new user, so the vote from a certified CIBUS Blogger will have more priority.
- Another interesting part of CIBUS voting system is stake based voting power. If a user carries more CIBUS Tokens then their vote will have more weightage than lower stakeholders.

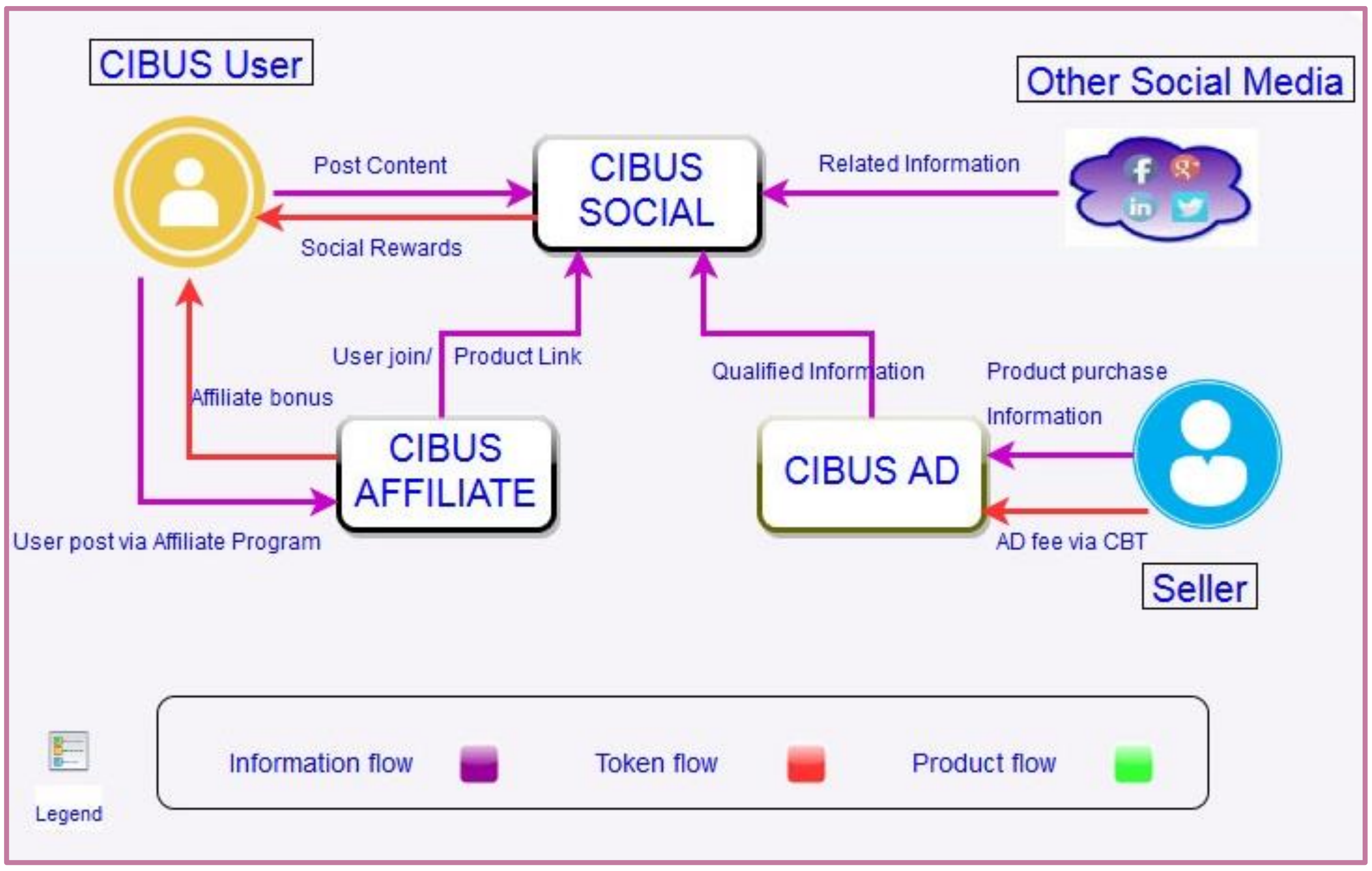


figure: CIBUS Social

3.3 CIBUS Retail

Under CIBUS Retail CIBUS mission is to provide a global online marketplace where practically anyone can trade in a trustworthy and transparent environment. We strive to offer our users the best available selection, and the utmost convenience with the promise of an attractive e-commerce service to satisfy customers' needs which is further bolstered by CIBUS escrow.

- An E-commerce platform for making direct sales to end consumer and to other businesses.
- Platform uses embedded smart contracts to conduct all transactions.
- An interactive price comparison screen that compares the price of all available alternatives for a given product from different websites.
- One click automated purchase strategy screen that shows the cheapest, nearest product based on the location preference set by the customer.
- Customers can set location preference to buy products from local, national or global sellers.
- An incorruptible vendor rating mechanism built using blockchain technology to protect users.
- Option to read past user reviews in all major world languages.
- Integrates past user reviews on the product from CIBUS-Social.
- Products with lower ratings will be displayed at the very bottom of the product list- automatic relegation to help users make better choices.
- Seller info on the product list will display the information disclosure level attained by the seller on CIBUS-Trace. Higher the level, better the disclosure standard.
- Product list to show other "connected" users who have purchased the product in the past.
- Virtual currency exchange counter that converts any other cryptocurrency to CIBUS tokens for maintaining minimum wallet balance for transaction.
- Five stage escrow service options provided for every transaction, irrespective of transaction size.
- Live product tracking screen for each transaction that shows how far away the product is in terms of distance and days.
- In-built mechanism will be implemented to protect the seller from orders being placed by bots or foes.
- Stringent Seller scrutiny will be maintained to keep the trading platforms free of scammers and fraudsters.
- Lowest possible transaction fee computed based on individual transaction value.
- Highest protection of customer details to prevent direct spamming from the seller is ensured.

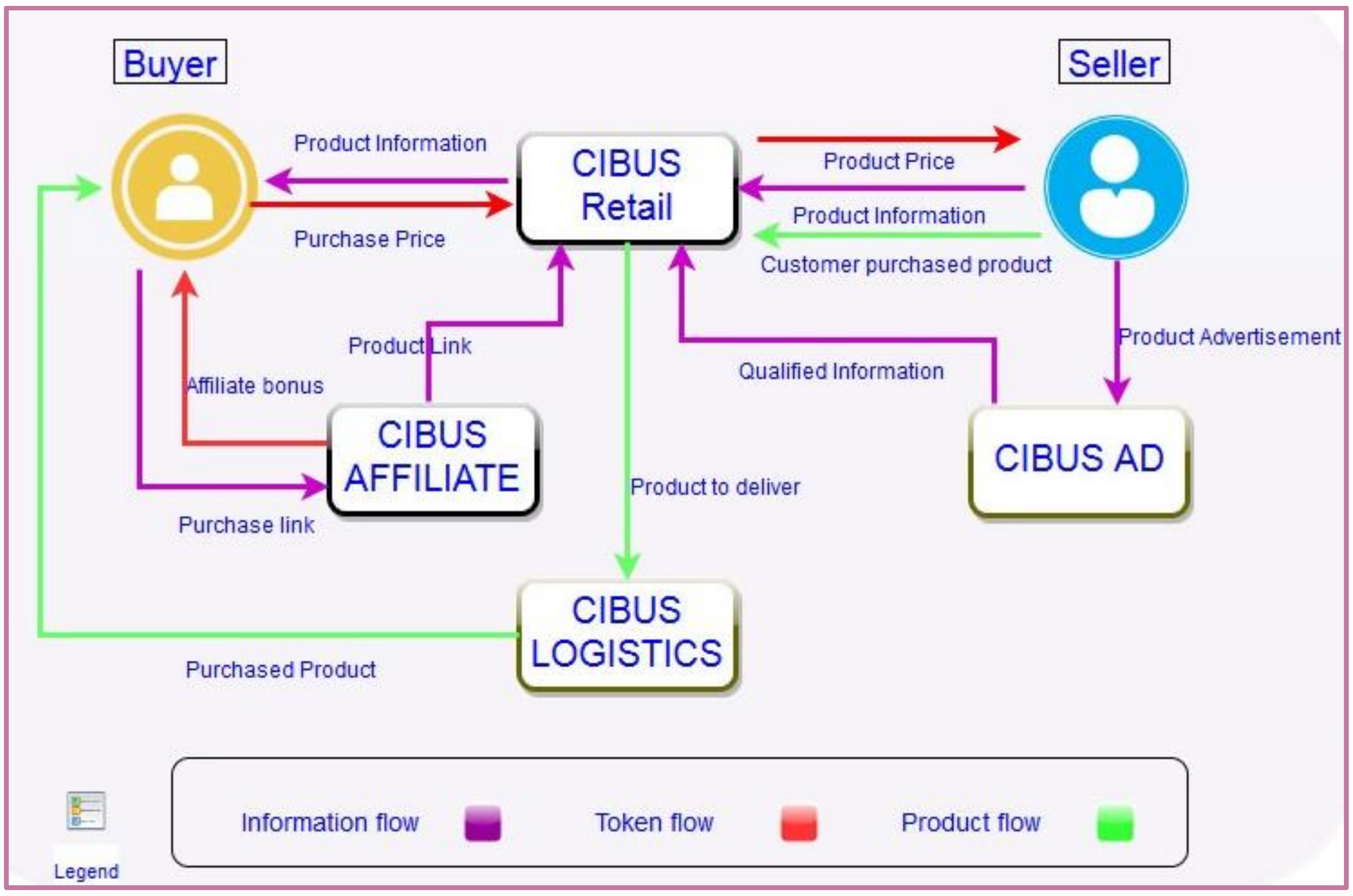


figure: CIBUS Retail

3.4 CIBUS Trade

CIBUS trade is a Business to business e-commerce platform for food products. CIBUS creates a win-all situation for the company and people like food bloggers, reviewers, chefs and many more interested parties to take advantage of CIBUS affiliate program and earn CIBUS token to increase their net worth. Some salient features described below makes CIBUS affiliate program even more robust which categorically consolidates CIBUS trade vertical.

Performance based - The main advantage of having an affiliate program is that it is entirely performance-based. Because affiliates are only paid a commission once the desired action has taken place, they're more motivated to drive the conversion you're looking for. This mitigates any efforts that drive traffic with little to no value to your company while also ensuring that you get what you pay for.

Broader Marketing Efforts - Affiliates can be found in every market and product category that exists today. The great news is that many of these affiliates will already have an established visitor base. These partnerships grant the opportunity to expand out into new markets that individuals might not otherwise have had the bandwidth to explore CIBUS thinks of these partners as an extension of its marketing strategy.

Validation - By partnering with trusted bloggers and reputable websites, product offering of CIBUS can further the reputation of the brands sold on its platform. This will further solidify consumer confidence in the product or service.

Cost Effective – CIBUS makes it cost effective by affiliate marketing. CIBUS is only paying commissions when the desired conversion occurs as per CIBUS policy. It saves cost of throwing away ad dollars on placements that have no proven value. Furthermore, large presence of food bloggers and chefs and food critics on internet makes it easy to associate affiliates.

Rapidly Scale Traffic - In conjunction with other marketing efforts, recruiting affiliates will allow scaling of traffic faster. The more sites that link to CIBUS pages, the more opportunities CIBUS will have to convert those users into paid customers. The other benefit of having these sites link back to CIBUS site is the added value search engines will put on the site. Essentially having outside resources link back to CIBUS site positively affects search engine ranking, which will compliment ICO efforts.

Users can shop for the products listed by manufacturers for sale by using CIBUS Retail sub-platform. An interactive price comparison platform will enable users to make informed choices. They can save the products on their cart and pay for it using CIBUS tokens. Users and manufacturers will be required to maintain a minimum wallet balance for initiating trading transactions. CIBUS will facilitate business to business trade by enabling direct selling and will also provide escrow services for a predetermined fee. Every transaction on CIBUS will attract a micro-fee based on the transaction size.

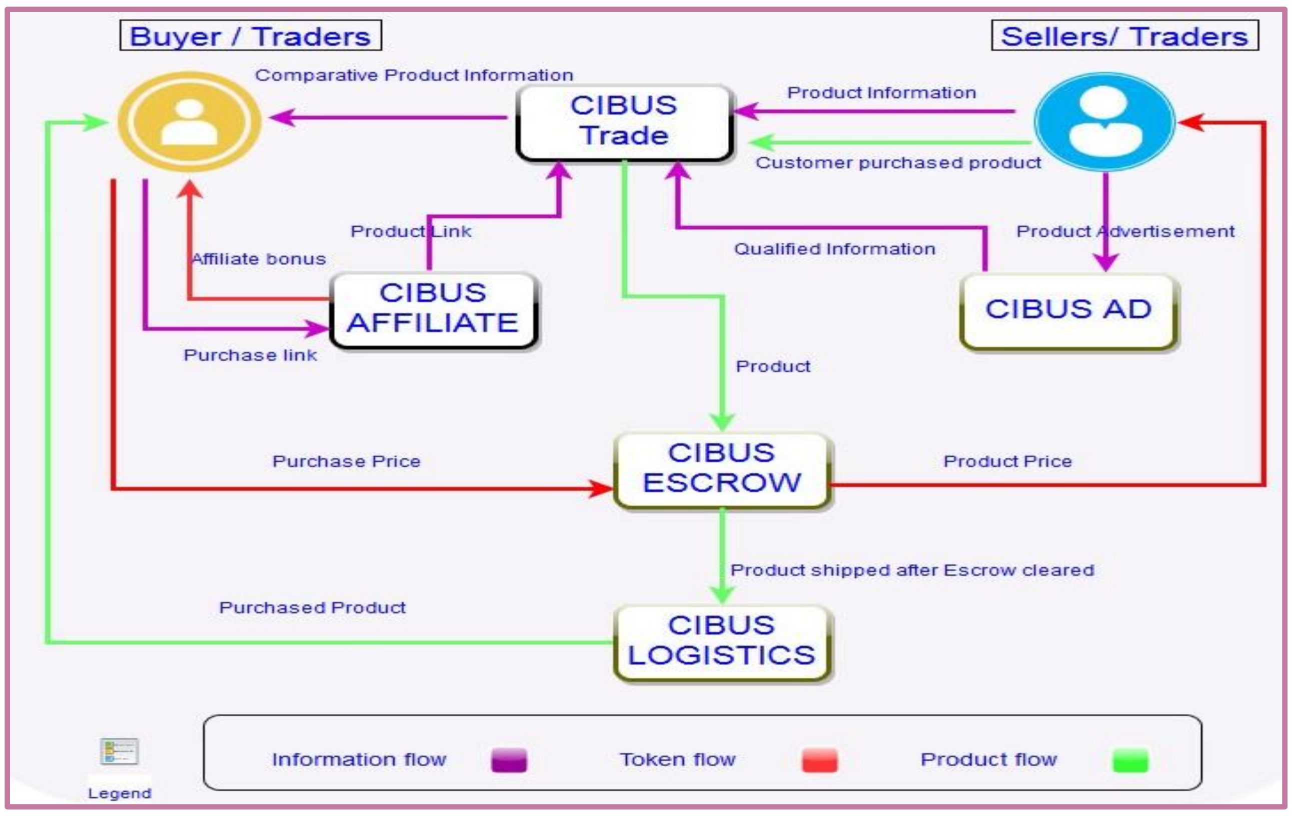


figure: CIBUS Trade

4. CIBUS PHASE 2

4.1 CIBUS Affiliate

CIBUS Affiliate service platform connects with leading bloggers and social media influencers in food and health supplement e - commerce trading network. CIBUS mission for affiliates is to help affiliate business community by boosting traffic and offering sales and protection for consumers from manipulation and food- scam.

<u>CIBUS Affiliation program</u>

1. CIBUS Affiliation is an affiliate marketing program that allows website owners and bloggers to create links and earn referral fees when customers click through and buy products from CIBUS E-Commerce platform. It's completely free to join.
2. It provides consumers the convenience of referring them to a CIBUS E Commerce site where they can immediately purchase products referrer advertise. When it is done, referrer can earn up to some portion in referral fees.
3. In the case of CIBUS Affiliation, referrer receives a commission when someone clicks on the special link and makes a qualifying purchase.

Protocols to follow: (included but not limited to):

- Customer should purchase the product within 24 hours' time frame.
- Cancellation of purchased product before delivery will not credit any money.
- No upper limit of number of links displayed in the page.
- No Duplicate/phishing link of CIBUS like URL to increase the hit count of your website, once identified will be banned for future association.
- User registration is mandatory before using the link in your page.
- It highly recommended to abide by CIBUS Affiliation policy.

Payment system:

1. Only a successful purchase according to CIBUS policy will ensure commission payment.
2. Payment will be done in form of CIBUS Token to the respective CIBUS affiliated crypto wallet of the referrer.

Benefits of CIBUS Affiliation Program:

- It's easy to understand and can be started right away: easy start makes it a great program for beginners.
- There are lots of useful products to promote, so referrer can yield benefit out of it.
- CIBUS ensures utmost reliance and provides quality products only.

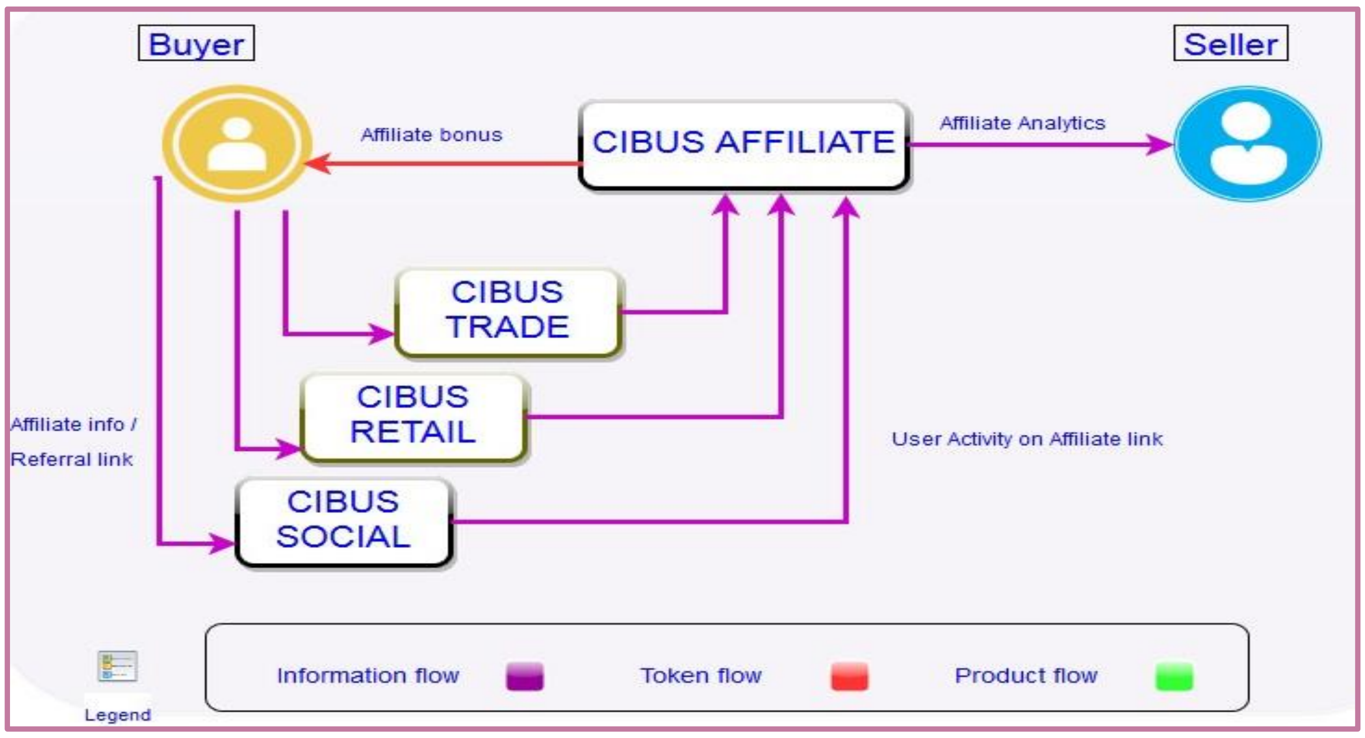


figure: CIBUS Affiliate

4.2 CIBUS AD

CIBUS AD facilitates direct targeted advertising by Manufacturers/producers by providing user preference and profile data. This will be the mainstay in CIBUS-AD.

The other features are:

- Eliminates intermediaries and allows direct marketing by the Manufacturers/Producers to the users.
- It is a decentralized advertising platform that is flexible to sellers needs and relevant to user interests.
- CIBUS AD integrates user profile and activity log from CIBUS Social, CIBUS Retail and CIBUS trade to provide valuable user demographic information to the manufacturers.
- CIBUS AD provides an inbuilt analytics platform- “CIBUS-Perspective” that provides rich and relevant data about user traffic on any manufacturer's profile page on CIBUS such as number of views and content relevance.
- Sellers can s create marketing content to target specific users based on their geo-location, or dietary requirements or age or gender or based on a combination of any of these factors.
- Manufacturers can create different advertisements for different types of users within a homogenous segment.
- CIBUS AD displays paid content at the top of any search results list on the CIBUS platform.
- CIBUS-Perspective at a basic level provides data on how much user engagement has been obtained for each advertisement, which content was viewed and which of it was skipped.
- Advertisement on CIBUS AD can be customized to link it back to the manufacturer's profile page on CIBUS or to their own websites.
- CIBUS AD supports video advertisements and will have an in-built “prevent AD-Skipping” feature for a few seconds.
- Stringent security measures to ensure that advertisements on CIBUS AD does not link the user to any malicious links, websites or ransomware
- CIBUS AD allows manufacturers to run a pilot advertisement campaign among selected users, before launching a full campaign. This enables the sellers to test the reach and effectiveness of their planned marketing campaign and affords them an opportunity to make modifications.
- CIBUS AD includes a performance evaluation tool that provides user metric data to gauge the success of the marketing campaigns.
- Advertisement space on CIBUS-AD will be sold on auction basis.
- CIBUS perspective and other user demographic data will be made available for a micro-fee.

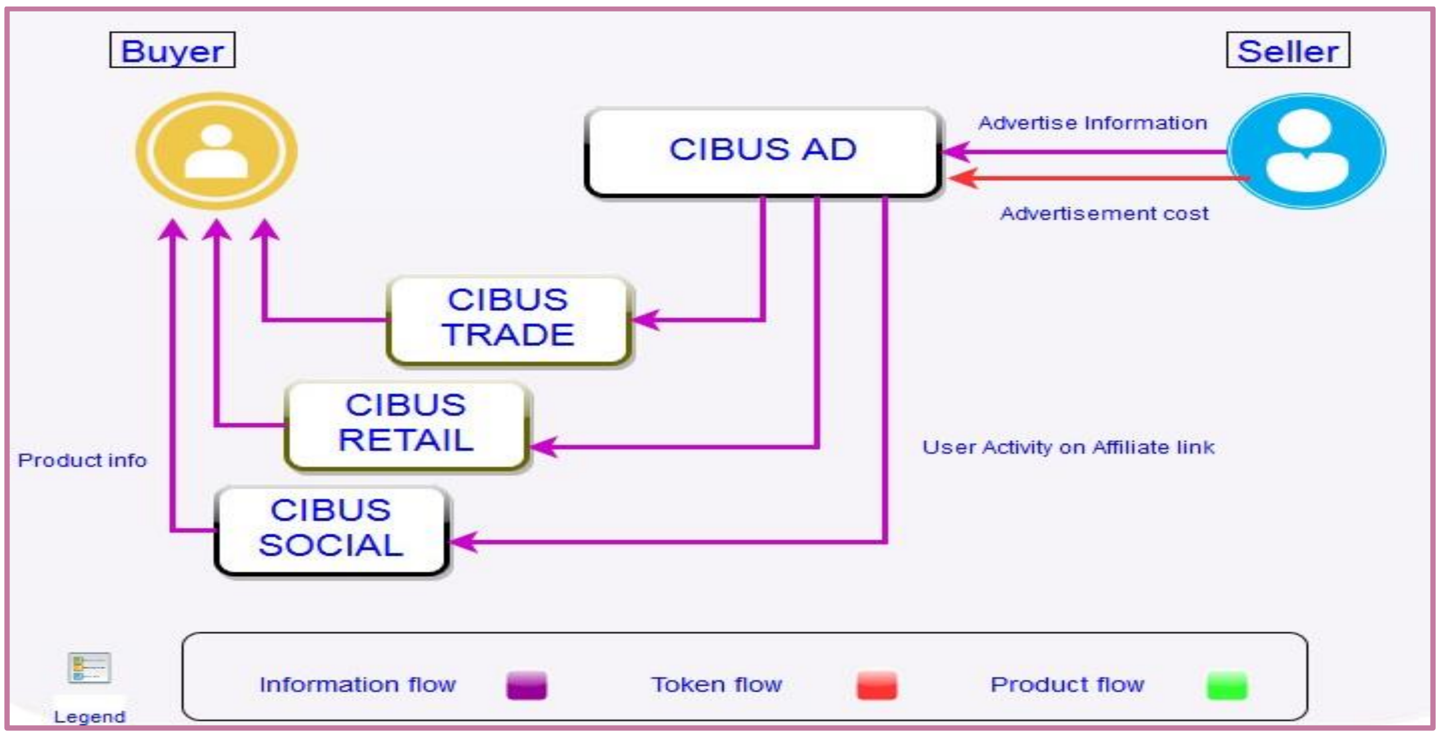


figure: CIBUS AD

4.3 CIBUS Escrow

CIBUS escrow service is provided against a fee, which is based on the transaction value on CIBUS platform

CIBUS-aim is to connect producers and traders of food and dietary supplements from all over the world and supply business with the essential tools and linked helps in the form of CIBUS escrow; CIBUS affiliates for introduction their produce to the global market with minimal transaction fees and without any stringent financial borders, intermediaries or annoying bureaucratic routine. CIBUS aims to unite food producers from all over the world in order to create a sovereign economical ecosystem.

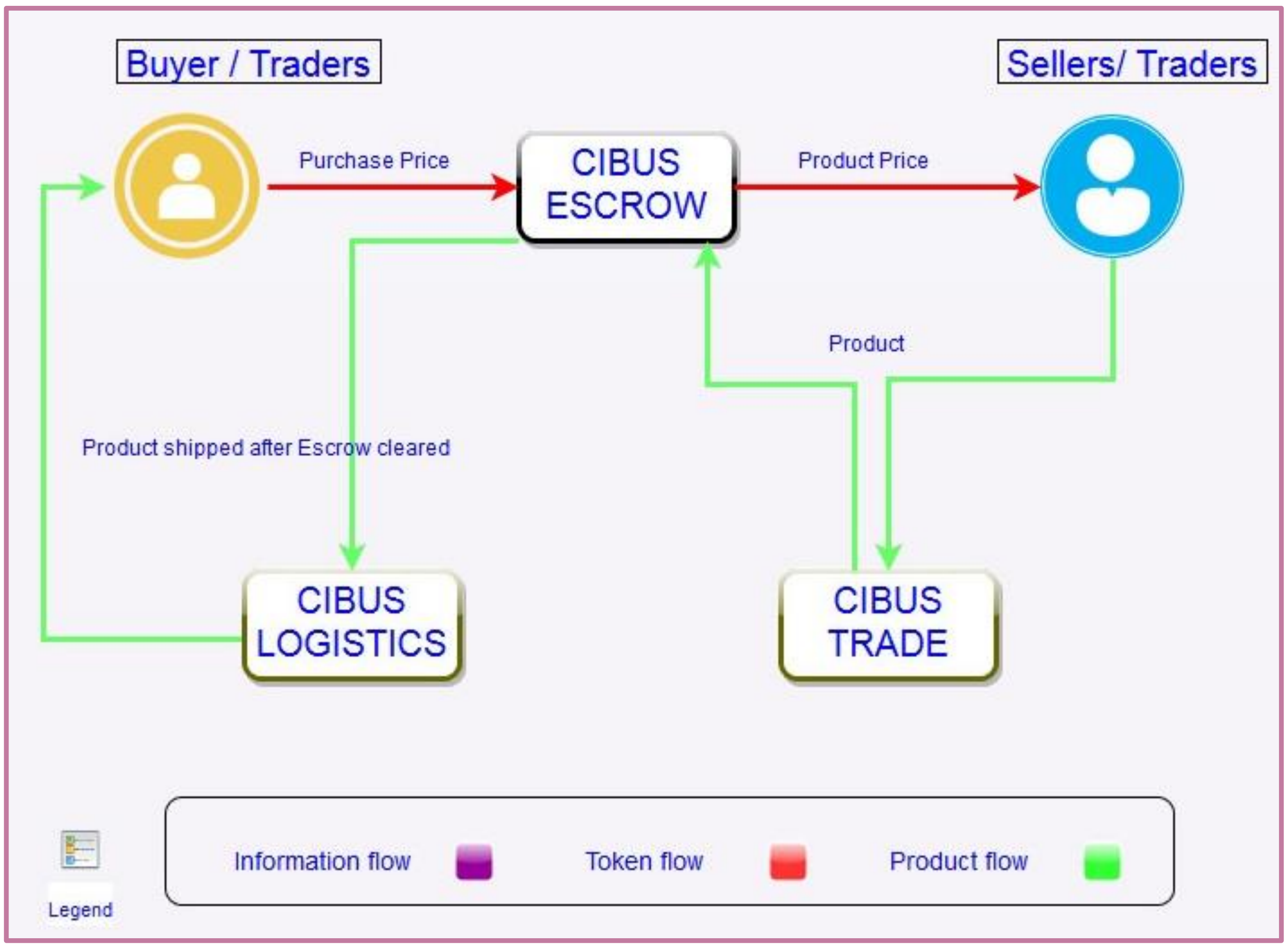


figure: CIBUS Escrow

4.4 CIBUS Logistics

CIBUS presents Global logistics solution that uses geo-tracking and stores conditions of transport on smart contracts to preserve the quality and integrity of food products and health supplements. By subscribing to CIBUS logistics platform, members will get to enjoy different facilities for shipping their food products by CIBUS, which will grow the reliability and efficiency of your products delivery at buyers' access.

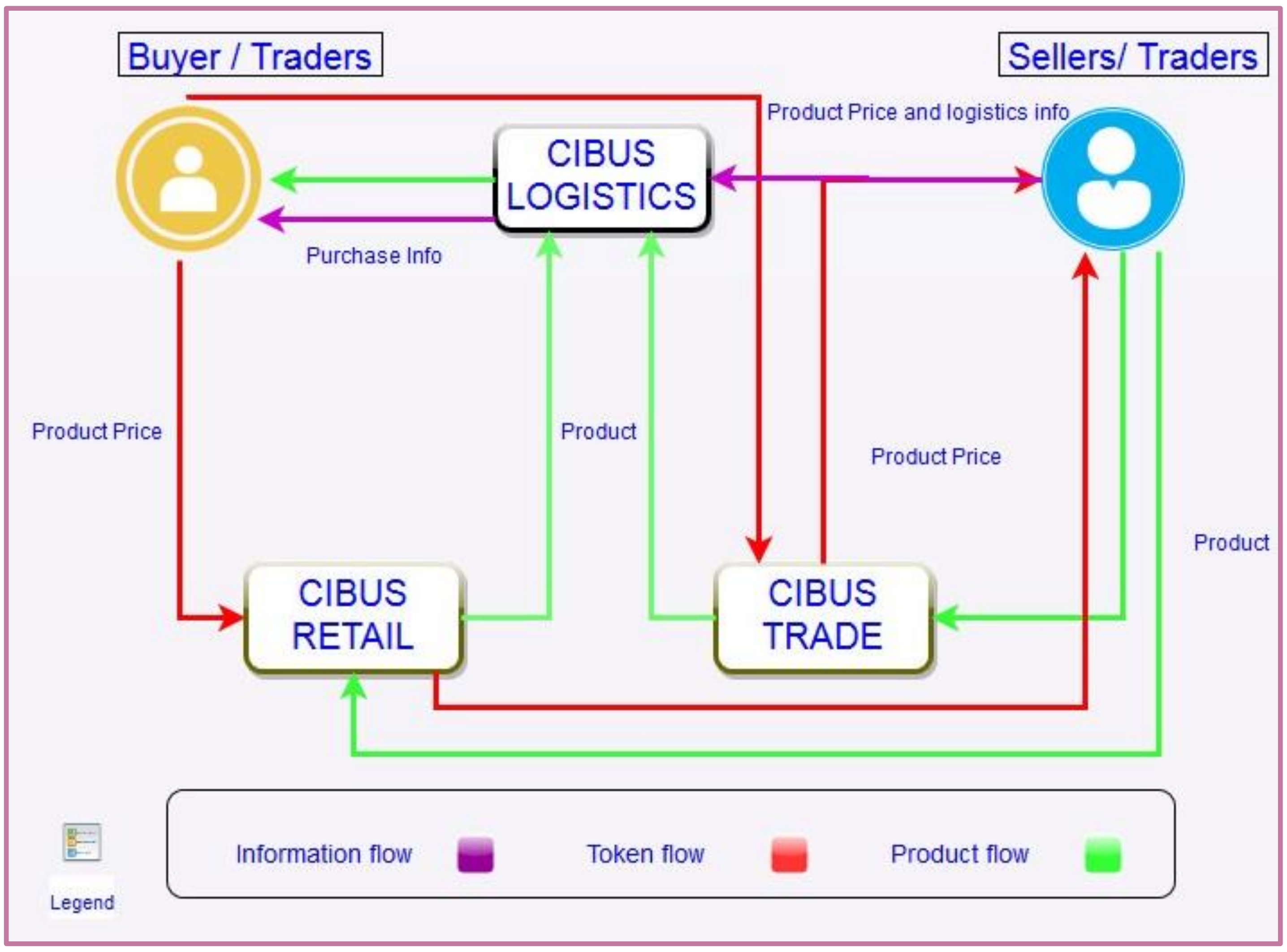


figure: CIBUS Logistics

5. CIBUS BLOCKCHAIN PROCESS

FOOD TRACEABILITY AND BLOCKCHAIN LEVERAGING PULL MODEL - The CIBUS Way

- **Push model** depicts the tracking of food life cycle which starts from Manufacturer and ends with Consumer.
- Transactional data and the parameters specified between the business parties will be recorded at each stage of the life cycle.
- Recording medium can be underlying Blockchain Network.
- Pull model defines the exact opposite flow of Push model.
- Here consumer will demand the information regarding the product; and the product business network will provide the data.
- Product business network can provide the full traceability information or no information or partial data regarding the product.
- Provided data will be stored in Blockchain network and will be supplied to customer on demand.
- So, the major difference is; our Application will not take part to store the traceable information at each stage of lifecycle of the food.
- Rather we will demand the information from manufacturer to provide traceable information and Blockchain will store the information as much as the manufacturer provides.

Manufacturer/Producer Perspective:

Producer will upload the following artefacts:

- Traceability information.
- Relevant documents
- Transactional information between different business parties involved
- Parameters of the product/Individual ledger information for each business party.

Standardized_Measurement Smart Contract:

- Latest standard in the market as per food safety rules will be there in the smart contract.
- After uploading the data and documents the smart contract will be executed and stored in Blockchain.
- Indicator will be stored for each parameter. (Passed or Failed)
- Stored information will be fetched on Customer demand.
- Document will be stored as a hash in the Blockchain for provenance.

Consumer Perspective:

Consumer demands all information pertaining to a product:

ConsumerDriven_Fetchall Smart Contract:

- It will fetch all information along with the safety indicators which was stored by producer.
- This will be executed when customer first time scans the QR code (or by other methods)

Consumer demands partial/filtered information pertaining to a product:

ConsumerDriven_Fetchpartial Smart Contract

- After getting all information, customer specifies certain criteria to see the product matches the food safety rules and /or having relevant information according to their need or not.
- This will be executed when customer already in the landing page of specific food related information.

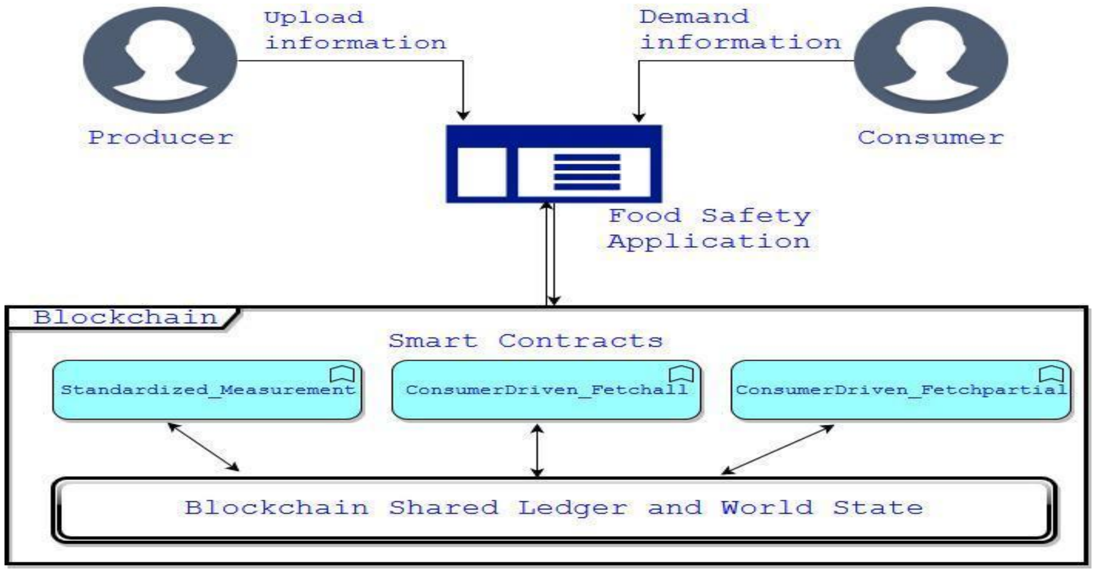


figure: CIBUS Food Trace System using Blockchain

Cibus Food Traceability workflow:

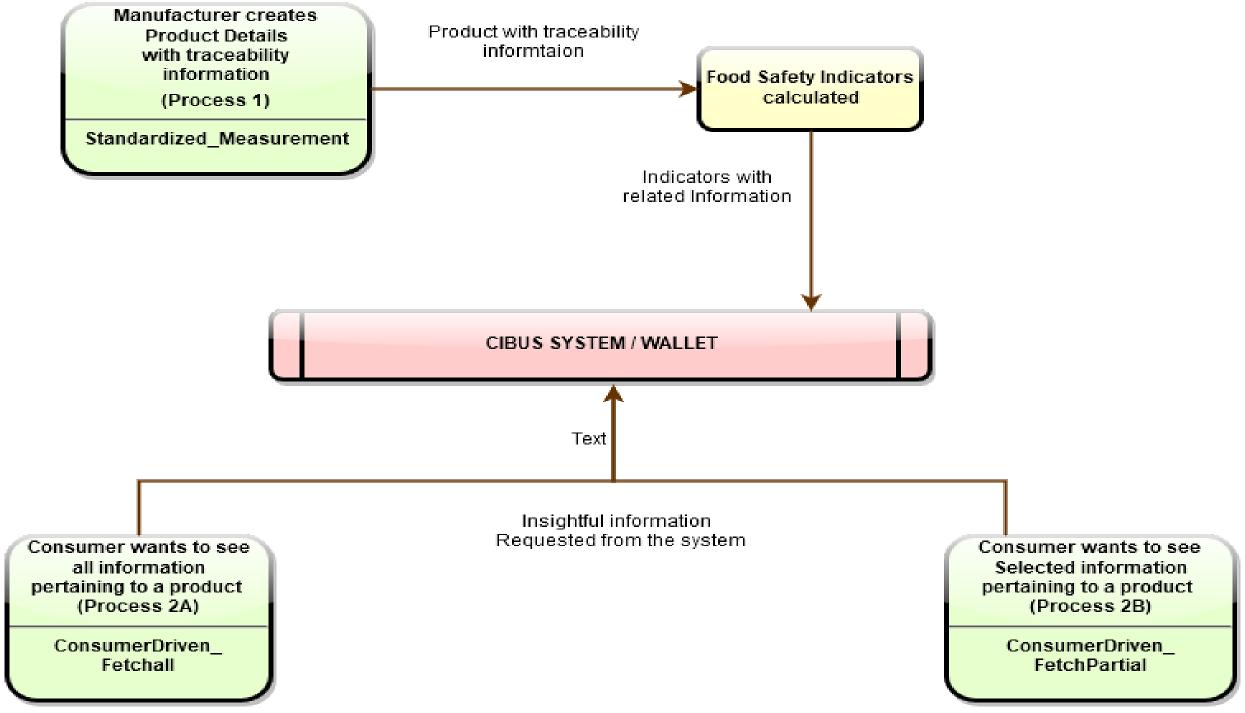


figure:Food trace pull model workflow using Blockchain

- Document storing inside blockchain

Documents cannot be stored directly inside Blockchain. When document is uploaded in the Blockchain, it applies one cryptographic algorithm (SHA256) and prepares one hash value, which is stored inside Blockchain DB. We can use different Database like DB2 or Oracle to store it.

This is useful for provenance perspective, as the same document will provide same hash always and minimal tampering of document will create totally different hash value. So, we can tally any two documents by comparing its hash value is same or not.

- CIBUS Blockchain in transparency and fraud prevention:

Transparency is the most demanding aspect of customers that has increased in recent years. The transparency has the ability to assess the performance of a supply chain and thereby increase the confidence of concerned players and of course improve the customer's trust. The strengths in the transparency of the blockchain lies in the trustworthiness as a transaction cannot be changed or manipulated afterwards. The trustworthiness cannot be achieved by a centralized system, since it is impossible for an outsider to assess the trustworthiness of the disclosed information. Therefore, it is advantageous to use a blockchain technology over a centralized system in terms of trustworthiness.

E-COMMERCE BUSINESS (RETAIL & TRADE VIA LOGISTICS) AND BLOCKCHAIN - THE CIBUS WAY

Ecommerce Platform and Escrow Service:

- An ecommerce platform is a software application that allows online businesses to manage their website, sales and operations.
- Fundamentally, ecommerce platforms are the backbone of any online retail enterprise, allowing the frontend and backend to work in tandem and efficiently. Ecommerce platforms are the girders of the ecommerce building, providing a rigid structure on which a B2B or B2C e-commerce enterprise is supported.
- Escrow service will enable the e commerce service provider to enable the capability which will hold customer payments until customers receives the goods that they purchased, and once customer confirm that he/she received the goods, Application will then release the funds to the seller.

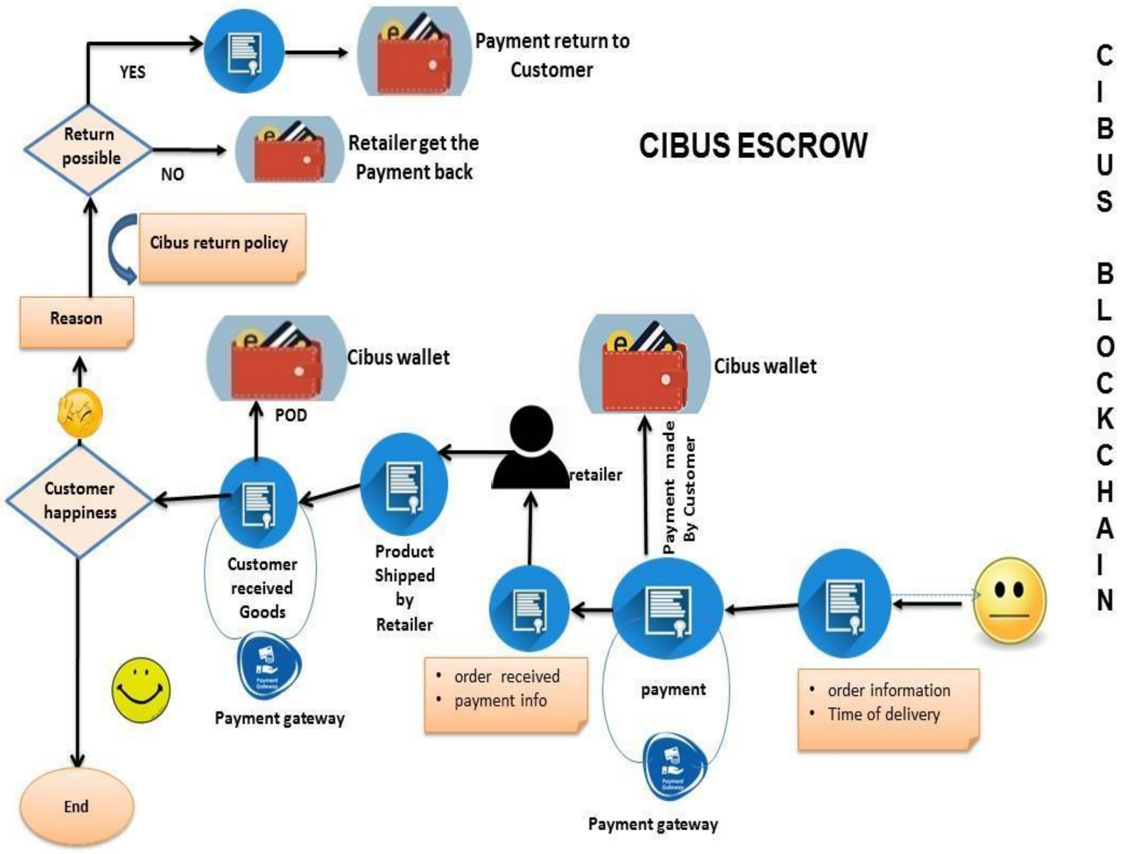


figure: CIBUS e-commerce business model

Blockchain Advantages for Ecommerce business:

- Platform to track order lifecycle.
- Provenance of each transaction.
- Speed of transaction and instant problem resolution opportunity.

From Blockchain perspective B2B and B2C business will have the similar methodology with minimal modification in data fields.

CIBUS platform broadly considers two types of Manufacture:

1. Manufacturer who conforms to trade/sell their goods by following Escrow service.
2. Manufacturer who are not ready to follow escrow service.

Manufacturer trading/selling workflow following Escrow service:

Customer places an order:

Customer_Order_ECOM Smart Contract will be executed:

1. The smart contract will insert the Order related information in Blockchain.
2. It will also calculate the estimate time of delivery and other information as per the contract/rule described in the Smart contract and replies back to customer.

Customer can pay instantly or opt for POD (Pay on delivery) option:

Customer_PayMode_ECOM

The smart contract will insert the Pay mode and price related information in Blockchain.

Customer_PayValue_ECOM (Conditional if customer done the payment):

- The smart contract will insert the payment related information in Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with CIBUS wallet/payment system.

CIBUS received the order and payment (conditional) and intimated the same information to Retailer:

Manufacturer_OrderReceived_ECOM

- The smart contract will insert the Order details along with pay mode and payment related information in Blockchain.
- It will also calculate the necessary details needed by Retailer and send the message to retailer.

Retailer shipped the product:

Retailer_OrderShipped_ECOM

The smart contract will insert the shipping related information in Blockchain.

Customer received the product and/or (conditional) avail the POD option.

Customer_OrderReceived_ECOM

The smart contract will insert the shipment date and other information in Blockchain.

Customer_PayValue_ECOM (Conditional if customer done the payment using POD):

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with CIBUS wallet/payment system.

(Conditional) Customer is unhappy with the product and ask for refund.

Customer_OrderRefund_ECOM

- Customer has to input some predefined values in terms of Question answer (for example using checkbox or Text Box) and have to include the problem details and reason for refund.
- CIBUS has some refund policies which will be made according to the business contract between CIBUS and registered retailer/traders, that contracts will be encoded in this smart contract.
- So, the smart contract will determine whether refund is possible according to the policies or not.

- If refund is possible then CIBUS application will call other smart contracts of the payment system to initiate the money refund. It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The incident will be reported to Retailer.

Customer_ReceiveValueRefund_ECOM

- The smart contract will insert the payment refund related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Customer wallet/payment system.

(Conditional) Retailer will receive the money: (If refund process not taken place)

Retailer_ReceiveValue_ECOM

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Retailer wallet/payment system.

(Conditional) Customer receive the refund amount: (If refund process taken place)

Retailer_PayValueRefund_ECOM

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Retailer wallet/payment system.

Customer_ReceiveValueRefund_ECOM

The smart contract will insert the payment refund related information Blockchain.

- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Customer wallet/payment system.

Manufacturer trading/selling workflow without following Escrow service:

Customer places an order:

Customer_Order_ECOM Smart Contract will be executed:

- The smart contract will insert the Order related information in Blockchain.
- It will also calculate the estimate time of delivery and other information as per the contract/rule described in the Smart contract and replies back to customer.

Customer can pay instantly or opt for POD (Pay on delivery) option:

Customer_PayMode_ECOM Smart Contract will be executed:

- The smart contract will insert the Paymode and price related information in Blockchain.

Customer_PayValue_ECOM Smart Contract will be executed (Conditional if customer done the payment):

- The smart contract will insert the payment related information in Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with CIBUS wallet/payment system.

Retailer_ReceiveValue_ECOM Smart Contract will be Executed: (Conditional if customer done the payment):

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Retailer wallet/payment system.

CIBUS received the order and payment (conditional) and intimated the same information to Retailer:

Manufacturer_OrderReceived_ECOM Smart Contract will be executed:

- The smart contract will insert the Order details along with paymode and payment related information in Blockchain.
- It will also calculate the necessary details needed by Retailer and send the message to retailer.

Retailer shipped the product:

Retailer_OrderShipped_ECOM Smart Contract will be executed:

- The smart contract will insert the shipping related information in Blockchain.

Customer received the product and/or (conditional) avail the POD option.

Customer_OrderReceived_ECOM Smart Contract will be executed:

The smart contract will insert the shipment date and other information in Blockchain.

Customer_PayValue_ECOM Smart Contract will be executed (Conditional if customer done the payment using POD):

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with CIBUS wallet/payment system.

Retailer_ReceiveValue_ECOM Smart Contract will be Executed: (Conditional if customer done the payment using POD):

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Retailer wallet/payment system.

(Conditional) Customer is unhappy with the product and ask for refund

Customer_OrderRefund_ECOM Smart Contract will be executed:

- Customer has to input some predefined values in terms of Question answer (for example using checkbox or Text Box) and have to include the problem details and reason for refund.
- CIBUS has some refund policies which will be made according to the business contract between CIBUS and registered retailer/traders, that contracts will be encoded in this smart contract.
- So, the smart contract will determine whether refund is possible according to the policies or not.
- If refund is possible then CIBUS application will call other smart contracts of the payment system to initiate the money refund. It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The incident will be reported to Retailer.

Customer will be intimated about refund status.

(Conditional) Customer receive the refund amount: (If refund process taken place)

Retailer_PayValueRefund_ECOM Smart Contract will be executed:

- The smart contract will insert the payment related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Retailer wallet/payment system.

Customer_ReceiveValueRefund_ECOM Smart Contract will be executed:

- The smart contract will insert the payment refund related information Blockchain.
- It will also interact with other smart contracts related to payment gateway, that will be described later and not in the scope of the document.
- The payment fund is now with Customer wallet/payment system.

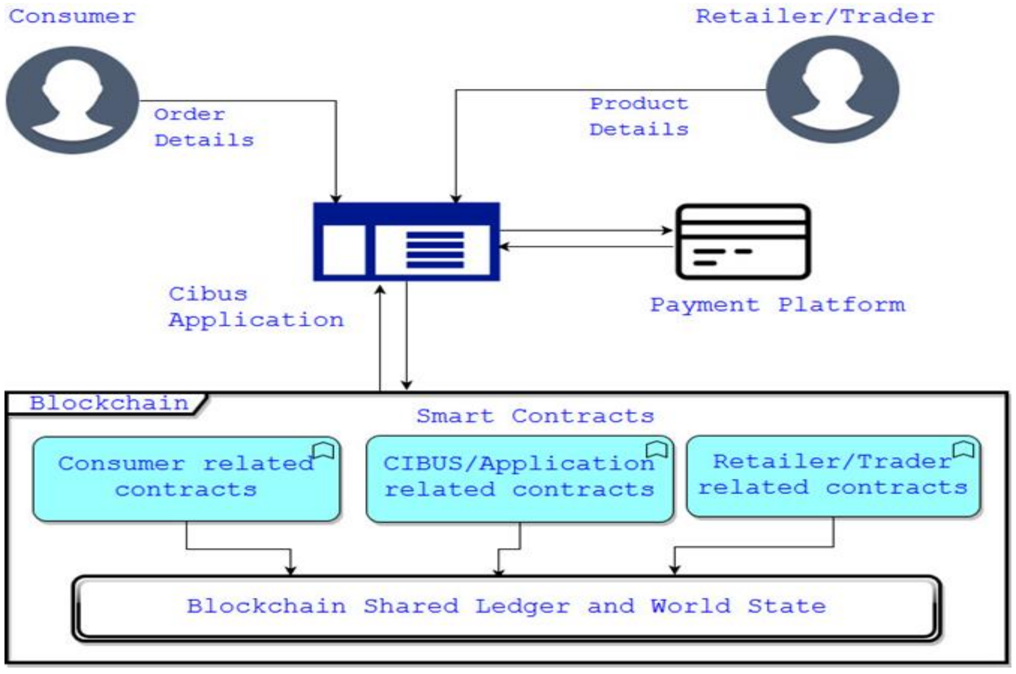

Consumer
Retailer/Trader
Order Details
Product Details
Cibus Application
Payment Platform
Blockchain
Smart Contracts
Consumer related contracts
CIBUS/Application related contracts
Retailer/Trader related contracts
Blockchain Shared Ledger and World State

figure : CIBUS e-commerce business interaction with Blockchain

Manufacturer trading/selling workflow following Escrow service:

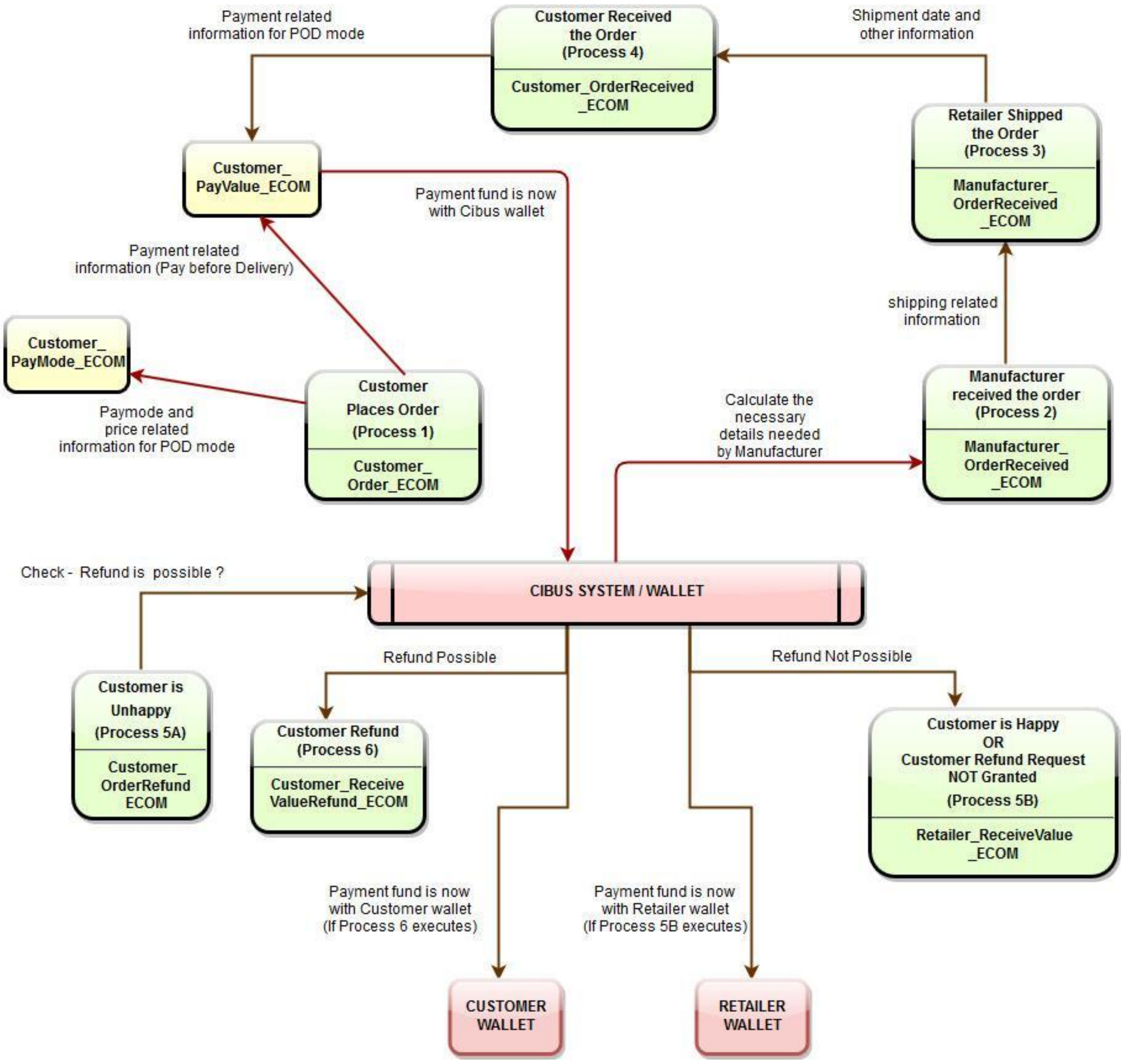

Payment related information for POD mode
Customer Received the Order (Process 4)
Customer_OrderReceived_ECOM
Shipment date and other information
Retailer Shipped the Order (Process 3)
Manufacturer_OrderReceived_ECOM
Customer_PayValue_ECOM
Payment fund is now with Cibus wallet
Payment related information (Pay before Delivery)
shipping related information
Customer_PayMode_ECOM
Customer Places Order (Process 1)
Customer_Order_ECOM
Paymode and price related information for POD mode
Calculate the necessary details needed by Manufacturer
Manufacturer received the order (Process 2)
Manufacturer_OrderReceived_ECOM
Check - Refund is possible ?
CIBUS SYSTEM / WALLET
Refund Possible
Refund Not Possible
Customer is Unhappy (Process 5A)
Customer_OrderRefund ECOM
Customer Refund (Process 6)
Customer_ReceiveValueRefund_ECOM
Customer is Happy OR Customer Refund Request NOT Granted (Process 5B)
Retailer_ReceiveValue_ECOM
Payment fund is now with Customer wallet (If Process 6 executes)
Payment fund is now with Retailer wallet (If Process 5B executes)
CUSTOMER WALLET
RETAILER WALLET

figure: CIBUS e-commerce workflow using Blockchain

Social Interaction and Blockchain - The CIBUS way

Blockchain Advantages for Social Interaction:

- One common platform to track interaction of customers among various dispersed Social platforms.
- Provenance of customer insight.
- Integration advantage between product specification/artefacts and social sentiment.

Smart Contracts:

Manufacturer/Producer Perspective:

Product_Information_Manufacturer_Social Smart Contract:

- The contract will insert the values in Blockchain shared ledger and update the world state by linking the product details already available in the Blockchain network.

- If it is a new product then it will call Standerdized_Measurement smart contract and add the details as other information part by creating a unique product information with a notification to Product owner to provide necessary information in compliance with generic product information Data structure available in the Blockchain network.

Product_Conversation_CIBUSPlatform_Manufacturer_Social Smart Contract:

- The contract will insert the values in Blockchain shared ledger and update the world state by linking the product details already available in the Blockchain network.
- If it is for new product then it will call Standerdized_Measurement smart contract and add the details as other information part by creating a unique product information with a notification to Product owner to provide necessary information in compliance with Generic product information Data structure available in the Blockchain network.

Stored information will be fetched on Customer demand. Document will be **stored as a hash** in the Blockchain for **provenance**.

Consumer Perspective:

Product_Conversation_CIBUSPlatform_Consumer_Social Smart Contract:

- The contract will insert the values in Blockchain shared ledger and update the world state by linking the product details already available in the Blockchain network.

Product_Conversation_OtherSocailPlatform_Consumer_Social Smart Contract:

- The contract will insert the values in Blockchain shared ledger and update the world state by fetching the social information of the relevant product given by registered CIBUS user for the product details already available in the Blockchain network.

ConsumerDriven_Fetchpartial Smart Contract:

- Consumer demands partial/filtered information pertaining to a product; then the smart Contract named will be executed. It has the same functionality as described earlier in food traceability. If the filter option is chosen as social media then application will show necessary information pertaining to that information.

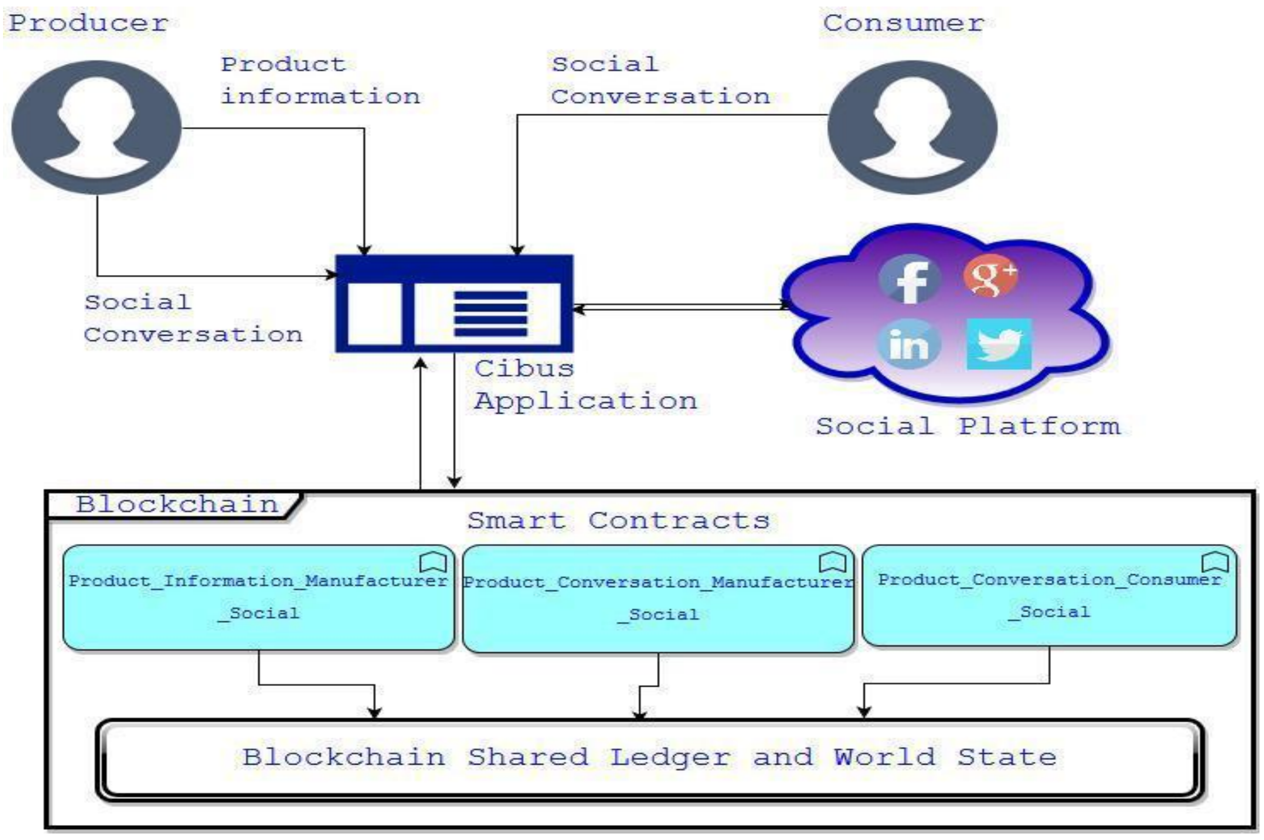

Producer
Product information
Social Conversation
Consumer
Social Conversation
Cibus Application
Social Platform
Blockchain
Smart Contracts
Product_Information_Manufacturer _Social
Product_Conversation_Manufacturer _Social
Product_Conversation_Consumer _Social
Blockchain Shared Ledger and World State

figure : Social Interaction using Blockchain

Social Incentive and Blockchain - The CIBUS way

Blockchain Advantages for Social Incentive:

- Blockchain platform is of most ease and amiable way to incentivize a user with CIBUS crypto token.
- Honest Review determination by CIBUS Users in a Decentralized system without any 3rd party influence.
- Provenance of user activities.
- Inbuilt smart contract advantage helps to calculate social incentive and send to user wallet simultaneously.

Smart Contracts:

Product_ConversationAuthenticity_Consumer_Social

Smart Contract will be executed to determine review/post/blog authenticity.

Product_ConversationFetch_CIBUSPlatform_Consumer_Social

Smart Contract will be executed to fetch the related review/post from CIBUS social platforms of the registered user and then the Smart Contract mentioned in point A will be executed to check the authenticity.

Product_ConversationFetch_OtherPlatform_Consumer_Social

Smart Contract will be executed to fetch the related review/post from the other social platforms of the registered user and then the Smart Contract mentioned in point A will be executed to check the authenticity.

Product_Conversation_IncetiveCalculation_Consumer_Social

One smart contract will be executed for each of the incentive calculation schemes (when applicable for example one of the applicable criteria – honest review). Note: Incentive calculation schemes are mentioned in 3rd point of 1st section.

Product_Conversation_BadgeIssue_Consumer_Social

Smart contract will be executed to issue recognized CIBUS badges to the registered users.

Product_Conversation_SendIncentive_Consumer_Social

Smart contract will be executed to send CIBUS Token to the user's wallet. This smart contract will take the input from incentive calculation smart contract and transfer the token.

Social workflow following Incentivize mechanism:

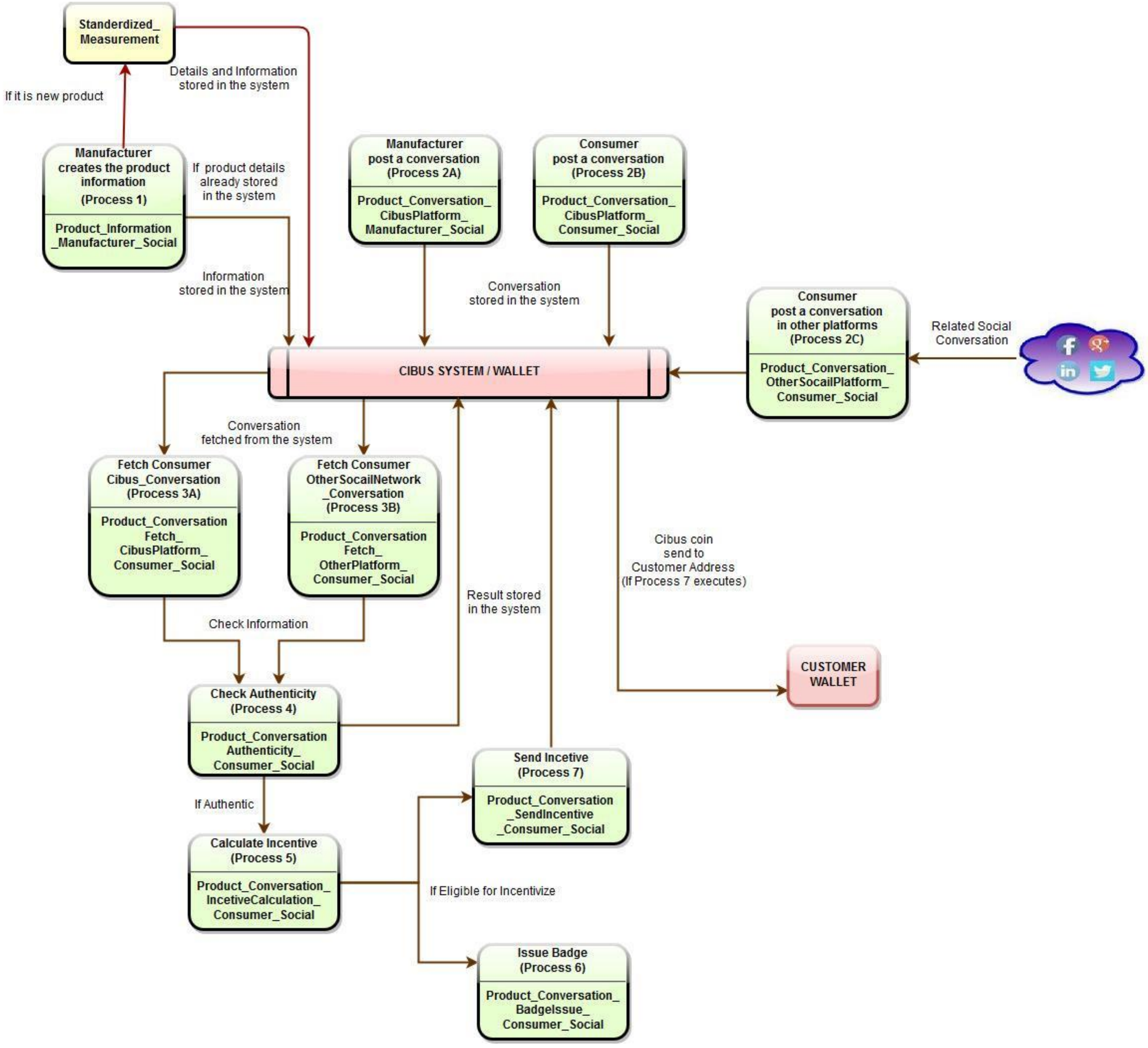

Standerdized_ Measurement
If it is new product
Details and Information stored in the system
Manufacturer creates the product information (Process 1)
Product_Information _Manufacturer_Social
If product details already stored in the system
Information stored in the system
Manufacturer post a conversation (Process 2A)
Product_Conversation_ CibusPlatform_ Manufacturer_Social
Consumer post a conversation (Process 2B)
Product_Conversation_ CibusPlatform_ Consumer_Social
Conversation stored in the system
Consumer post a conversation in other platforms (Process 2C)
Product_Conversation_ OtherSocailPlatform_ Consumer_Social
Related Social Conversation
CIBUS SYSTEM / WALLET
Conversation fetched from the system
Fetch Consumer Cibus_Conversation (Process 3A)
Product_Conversation Fetch_ CibusPlatform_ Consumer_Social
Fetch Consumer OtherSocailNetwork _Conversation (Process 3B)
Product_Conversation Fetch_ OtherPlatform_ Consumer_Social
Check Information
Result stored in the system
Cibus coin send to Customer Address (If Process 7 executes)
CUSTOMER WALLET
Check Authenticity (Process 4)
Product_Conversation Authenticity_ Consumer_Social
If Authentic
Send Incetive (Process 7)
Product_Conversation _SendIncentive _Consumer_Social
Calculate Incentive (Process 5)
Product_Conversation_ IncetiveCalculation_ Consumer_Social
If Eligible for Incentivize
Issue Badge (Process 6)
Product_Conversation_ BadgeIssue_ Consumer_Social

figure : Social workflow using Blockchain

AFFILIATION PROGRAM VIA E COMMERCE AND BLOCKCHAIN - THE CIBUS WAY

Blockchain Advantages for Affiliation Program:

- Blockchain platform is a simple and amicable way to provide Affiliation program which comprises easy tracking of the customer activity and giving away the payment.
- Provenance of user activities. It will help to get insight of product popularity information.
- In-built smart contract advantage will help in calculating referral bonus and send to user wallet simultaneously.

Smart Contracts:

1. One smart contract will be executed for each of the link creation activity by user.
2. One smart contract will be executed to track the customer activity from clicking the link to purchase lifecycle.
3. One smart contract will be executed to calculate the payment according to the CIBUS affiliation rules to send CIBUS Token to the user's wallet. This smart contract will take the input from customer activity (link lifecycle) smart contract and transfer the token.

ADVERTISEMENT PROGRAM AND BLOCKCHAIN - THE CIBUS WAY

Blockchain Advantages for AD Program:

- Log user activity from different platforms like CIBUS Social, CIBUS Retail and CIBUS trade in a common shared ledger to provide insightful user activities.
- Log User activity on posted advertisement.
- Business contract/agreement between Advertiser and CIBUS can be possible via Smart contract.
- Easy Advertisement posting payment in terms of Cryptocurrency via Smart contract functionality.

Smart Contracts:

Advertisement_CIBUSADPlatform_AD

The smart contract logs the advertisement information request by the user with the content details in the shared ledger.

AdContentVerify_CIBUSADPlatform_AD

The smart contract will verify the content and user intention adhere to CIBUS Ad protocol or not.

UserActivity_CIBUSPlatform_AD

The smart contract will inspect the user activities based on certain parameters needful for advertising; from other CIBUS platforms like CIBUS Social to improve the advertising content to be posted by Advertiser

UserActivity_CIBUSADPlatform_AD

The smart contract will inspect the performance of the user reaction (like seeing the full advertisement or clicking any link associated with the URL) and logs it in the shared ledger. It will help to get micro level user activities and help to build a CIBUS Analytics.

BusinessAgreement_CIBUSADPlatform_AD

The smart contract will execute the business agreement encoded in Blockchain smart contract programming language.

Payment_CIBUSADPlatform_AD

The smart contract will execute the agreement of payment details between CIBUS and advertiser. It will also interact with other payment related smart contracts to transfer the fees from one wallet to another wallet.

Cibus Advertisement workflow:

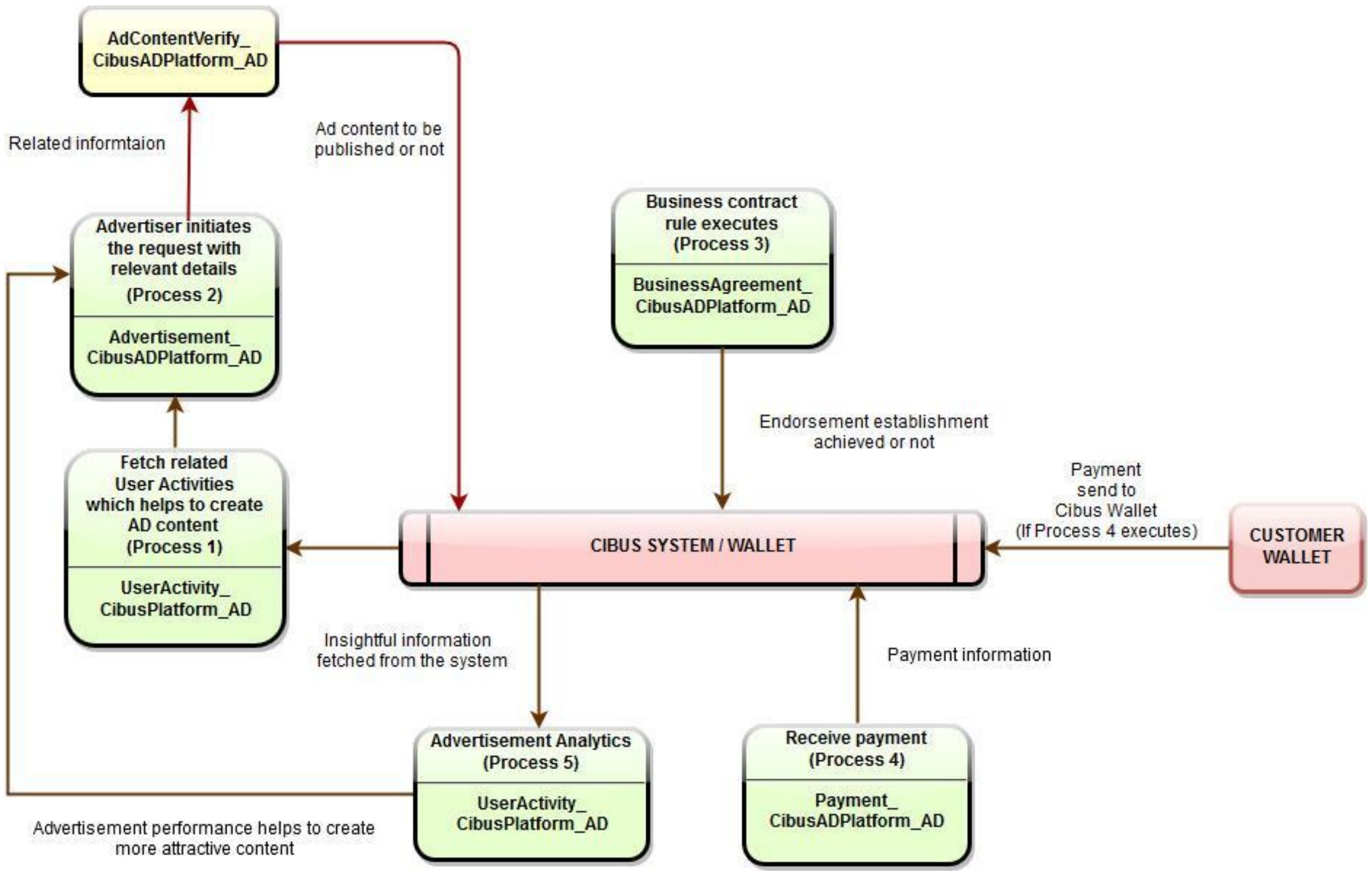
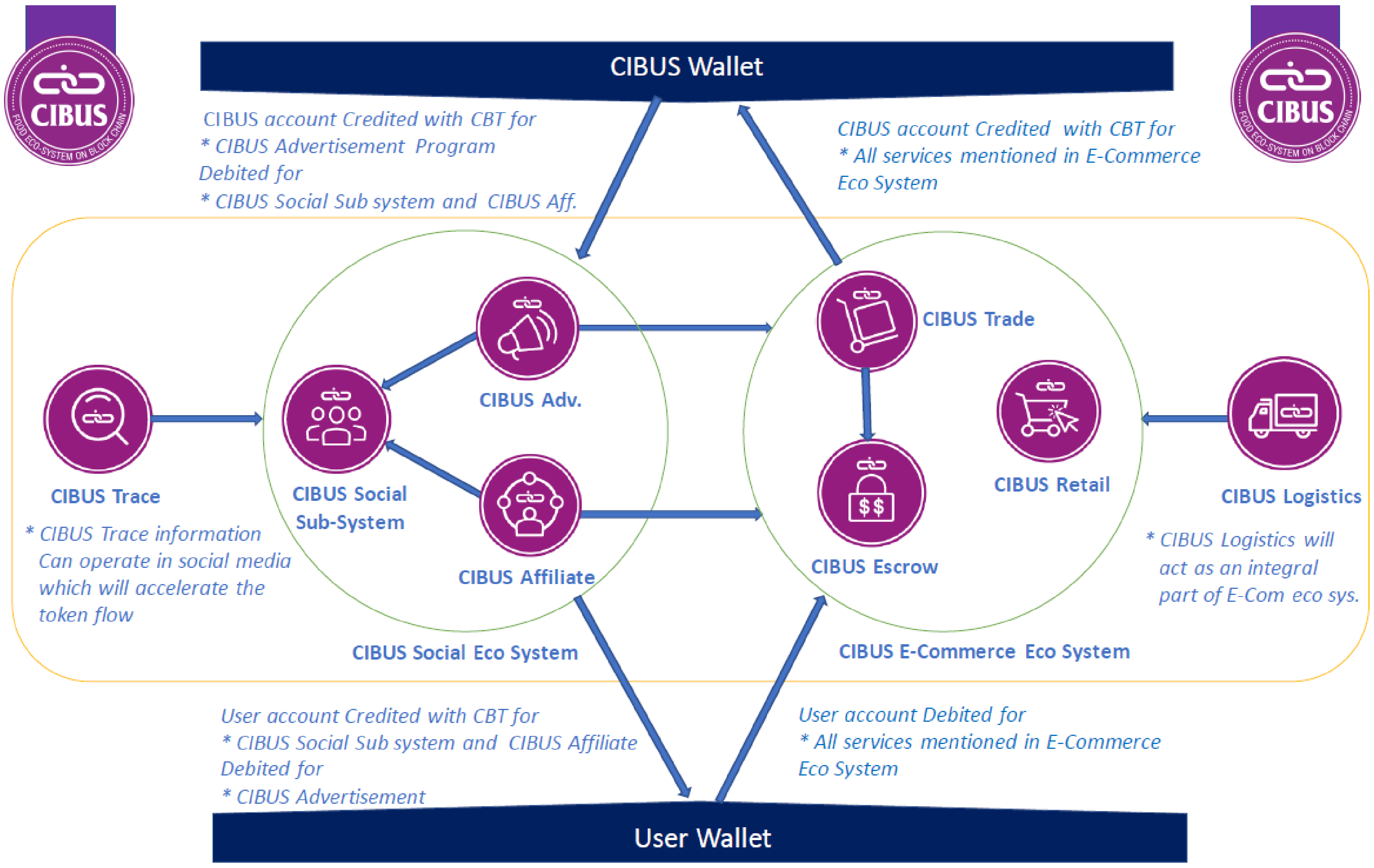


figure : Advertisement workflow using Blockchain

TOKEN FLOW INSIDE CIBUS ECOSYSTEM:

CIBUS Token is the financial component which will be used to interact with each vertical of CIBUS ecosystem. CIBUS application users and business parties associated with CIBUS system can participate in all functional processes.



CIBUS account Credited with CBT for
* CIBUS Advertisement Program
Debited for
* CIBUS Social Sub system and CIBUS Aff.

CIBUS account Credited with CBT for
* All services mentioned in E-Commerce Eco System

* CIBUS Trace information Can operate in social media which will accelerate the token flow

* CIBUS Logistics will act as an integral part of E-Com eco sys.

User account Credited with CBT for
* CIBUS Social Sub system and CIBUS Affiliate
Debited for
* CIBUS Advertisement

User account Debited for
* All services mentioned in E-Commerce Eco System

figure: CIBUS Token flow

6. CIBUS TOKEN DETAILS

About: This section provides the source code and testing of CIBUS Token smart contract file.

Contract Information:

1. **Contract Details:**
   Contract type: ERC-20 standard Token
   Token name: CIBUS Token
   Token Symbol: CBT
   Decimal points accepted: 10

2. **Contract Source Code:**

```
pragma solidity ^0.4.19;

/*
 * Abstract Token Smart Contract.  Copyright © 2017 by CIBUS WORLD.
 * Author: contact@cibus.world
 */


 /*
 * Safe Math Smart Contract.  Copyright © 2017-2018 by CIBUS WORLD.
 * Author: contact@cibus.world
 * https://github.com/OpenZeppelin/zeppelin-
solidity/blob/master/contracts/math/SafeMath.sol
 */

contract SafeMath {
  function mul(uint256 a, uint256 b) internal pure returns (uint256) {
     if (a == 0) {
     return 0;
     }
     uint256 c = a * b;
     assert(c / a == b);
     return c;
  }

  function safeDiv(uint256 a, uint256 b) internal pure returns (uint256) {
     // assert(b > 0); // Solidity automatically throws when dividing by 0
     uint256 c = a / b;
     // assert(a == b * c + a % b); // There is no case in which this doesn't hold
     return c;
  }

  function safeSub(uint256 a, uint256 b) internal pure returns (uint256) {
     assert(b <= a);
     return a - b;
  }

  function safeAdd(uint256 a, uint256 b) internal pure returns (uint256) {
     uint256 c = a + b;
     assert(c >= a);
     return c;
  }
}




/**
 * ERC-20 standard token interface, as defined
 * <a href="http://github.com/ethereum/EIPs/issues/20">here</a>.
 */
contract Token {

  function totalSupply() constant returns (uint256 supply);
  function balanceOf(address _owner) constant returns (uint256 balance);
  function transfer(address _to, uint256 _value) returns (bool success);
  function transferFrom(address _from, address _to, uint256 _value) returns (bool
success);
  function approve(address _spender, uint256 _value) returns (bool success);
  function allowance(address _owner, address _spender) constant returns (uint256
remaining);
  event Transfer(address indexed _from, address indexed _to, uint256 _value);
  event Approval(address indexed _owner, address indexed _spender, uint256 _value);
}



/**
 * Abstract Token Smart Contract that could be used as a base contract for
 * ERC-20 token contracts.
 */
contract AbstractToken is Token, SafeMath {
  /**
   * Create new Abstract Token contract.
   */
  function AbstractToken () {
     // Do nothing
  }

  /**
   * Get number of tokens currently belonging to given owner.
   *
   * @param _owner address to get number of tokens currently belonging to the
   *      owner of
   * @return number of tokens currently belonging to the owner of given address
   */
  function balanceOf(address _owner) constant returns (uint256 balance) {
     return accounts [_owner];
  }

  /**
   * Transfer given number of tokens from message sender to given recipient.
   *
   * @param _to address to transfer tokens to the owner of
   * @param _value number of tokens to transfer to the owner of given address
   * @return true if tokens were transferred successfully, false otherwise
   * accounts [_to] + _value > accounts [_to] for overflow check
   * which is already in safeMath
   */
  function transfer(address _to, uint256 _value) returns (bool success) {
     require(_to != address(0));
     if (accounts [msg.sender] < _value) return false;
     if (_value > 0 && msg.sender != _to) {
     accounts [msg.sender] = safeSub (accounts [msg.sender], _value);
     accounts [_to] = safeAdd (accounts [_to], _value);
     }
     Transfer (msg.sender, _to, _value);
     return true;
  }

  /**
   * Transfer given number of tokens from given owner to given recipient.
   *
   * @param _from address to transfer tokens from the owner of
   * @param _to address to transfer tokens to the owner of
   * @param _value number of tokens to transfer from given owner to given
   *      recipient
   * @return true if tokens were transferred successfully, false otherwise
   * accounts [_to] + _value > accounts [_to] for overflow check
   * which is already in safeMath
   */
  function transferFrom(address _from, address _to, uint256 _value)
  returns (bool success) {
     require(_to != address(0));
     if (allowances [_from][msg.sender] < _value) return false;
     if (accounts [_from] < _value) return false;

     if (_value > 0 && _from != _to) {
      allowances [_from][msg.sender] = safeSub (allowances [_from][msg.sender],
_value);
     accounts [_from] = safeSub (accounts [_from], _value);
     accounts [_to] = safeAdd (accounts [_to], _value);
     }
     Transfer(_from, _to, _value);
     return true;
  }

  /**
   * Allow given spender to transfer given number of tokens from message sender.
   * @param _spender address to allow the owner of to transfer tokens from message
sender
   * @param _value number of tokens to allow to transfer
   * @return true if token transfer was successfully approved, false otherwise
   */
   function approve (address _spender, uint256 _value) returns (bool success) {
     allowances [msg.sender][_spender] = _value;
     Approval (msg.sender, _spender, _value);
     return true;
  }

  /**
   * Tell how many tokens given spender is currently allowed to transfer from
   * given owner.
   *
   * @param _owner address to get number of tokens allowed to be transferred
   *      from the owner of
   * @param _spender address to get number of tokens allowed to be transferred
   *      by the owner of
   * @return number of tokens given spender is currently allowed to transfer
   *      from given owner
   */
  function allowance(address _owner, address _spender) constant
  returns (uint256 remaining) {
     return allowances [_owner][_spender];
  }

  /**
   * Mapping from addresses of token holders to the numbers of tokens belonging
   * to these token holders.
   */
  mapping (address => uint256) accounts;

  /**
   * Mapping from addresses of token holders to the mapping of addresses of
   * spenders to the allowances set by these token holders to these spenders.
   */
  mapping (address => mapping (address => uint256)) private allowances;
}


/**
 * CIBUS token smart contract.
 */
contract CIBUSToken is AbstractToken {
  /**
   * Maximum allowed number of tokens in circulation.
   *Total Supply for Pre ICO = (10% of 100000000 = 10000000)  and  ICO = (30% of
100000000 = 30000000)
   * 10^^10 is done for decimal places, this is standard practice as all ethers are
actually wei in EVM
   */


  uint256 constant MAX_TOKEN_COUNT = 100000000 * (10**10);

  /**
   * Address of the owner of this smart contract.
   */
  address private owner;

  /**
   * Current number of tokens in circulation.
   */
  uint256 tokenCount = 0;


  /**
   * True if tokens transfers are currently frozen, false otherwise.
   */
  bool frozen = false;

  /**
   * Counter of total funds collected, in wei
  */
  uint public totalCollected = 0;
```

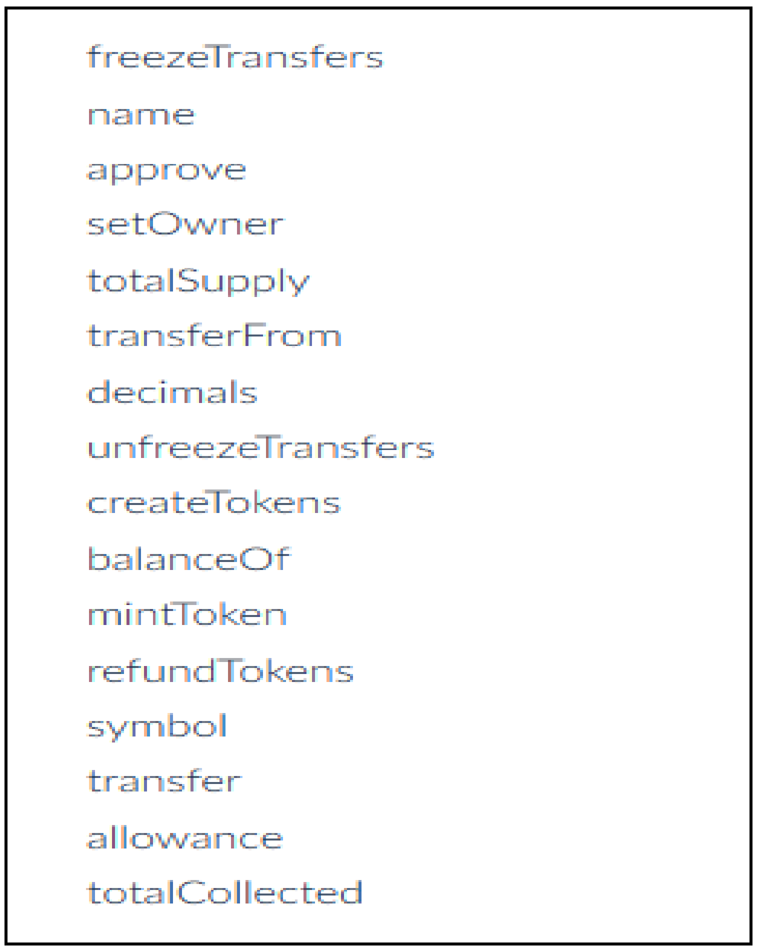
```
/**
 * Create new CIBUS token smart contract and make msg.sender the
 * owner of this smart contract.
 */
function CIBUSToken () {
   owner = msg.sender;
}

/**
 * Get total number of tokens in circulation.
 *
 * @return total number of tokens in circulation
 */
function totalSupply() constant returns (uint256 supply) {
   return tokenCount;
}

string constant public name = "CIBUS Token";
string constant public symbol = "CBT";
uint8 constant public decimals = 10;

/**
 * Transfer given number of tokens from message sender to given recipient.
 *
 * @param _to address to transfer tokens to the owner of
 * @param _value number of tokens to transfer to the owner of given address
 * @return true if tokens were transferred successfully, false otherwise
 */
function transfer(address _to, uint256 _value) returns (bool success) {
   if (frozen) return false;
   else return AbstractToken.transfer (_to, _value);
}

/**
 * Transfer given number of tokens from given owner to given recipient.
 *
 * @param _from address to transfer tokens from the owner of
 * @param _to address to transfer tokens to the owner of
 * @param _value number of tokens to transfer from given owner to given
 *      recipient
 * @return true if tokens were transferred successfully, false otherwise
 */
function transferFrom(address _from, address _to, uint256 _value)
   returns (bool success) {
   if (frozen) return false;
   else return AbstractToken.transferFrom (_from, _to, _value);
}

 /**
 * Change how many tokens given spender is allowed to transfer from message
 * spender.  In order to prevent double spending of allowance,
 * To change the approve amount you first have to reduce the addresses`
 * allowance to zero by calling `approve(_spender, 0)` if it is not
 * already 0 to mitigate the race condition described here:
 * https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
 * @param _spender address to allow the owner of to transfer tokens from
 *      message sender
 * @param _value number of tokens to allow to transfer
 * @return true if token transfer was successfully approved, false otherwise
 */
function approve (address _spender, uint256 _value)
   returns (bool success) {
  require(allowance (msg.sender, _spender) == 0 || _value == 0);
   return AbstractToken.approve (_spender, _value);
}

/**
 * Create _value new tokens and give new created tokens to msg.sender.
 * May only be called by smart contract owner.
 *
 * @param _value number of tokens to create
 * @param _collected total amounts of fund collected for this issuance, in wei
 * @return true if tokens were created successfully, false otherwise
 */
function createTokens(uint256 _value, uint _collected)
   returns (bool success) {
   require (msg.sender == owner);

   if (_value > 0) {
   if (_value > safeSub (MAX_TOKEN_COUNT, tokenCount)) return false;

   accounts [msg.sender] = safeAdd (accounts [msg.sender], _value);
   tokenCount = safeAdd (tokenCount, _value);
   totalCollected = safeAdd(totalCollected, _collected);

    // adding transfer event and _from address as null address
    Transfer(0x0, msg.sender, _value);

    return true;
   }

    return false;

}


/**
 * For future use only whne we will need more tokens for our main application
 * Create mintedAmount new tokens and give new created tokens to target.
 * May only be called by smart contract owner.
 * @param mintedAmount number of tokens to create
 * @return true if tokens were created successfully, false otherwise
 */

function mintToken(address target, uint256 mintedAmount)
returns (bool success) {
   require (msg.sender == owner);
   if (mintedAmount > 0) {

   accounts [target] = safeAdd (accounts [target], mintedAmount);
   tokenCount = safeAdd (tokenCount, mintedAmount);

    // adding transfer event and _from address as null address
    Transfer(0x0, target, mintedAmount);

   return true;
   }
    return false;

   }

/**
 * Set new owner for the smart contract.
 * May only be called by smart contract owner.
 *
 * @param _newOwner address of new owner of the smart contract
 */
function setOwner(address _newOwner) {
   require (msg.sender == owner);

   owner = _newOwner;
}

/**
 * Freeze token transfers.
 * May only be called by smart contract owner.
 */
function freezeTransfers () {
   require (msg.sender == owner);

   if (!frozen) {
   frozen = true;
   Freeze ();
   }
```

```
  }

  /**
   * Unfreeze token transfers.
   * May only be called by smart contract owner.
   */
  function unfreezeTransfers () {
     require (msg.sender == owner);

     if (frozen) {
     frozen = false;
     Unfreeze ();
     }
  }

  /*A user is able to unintentionally send tokens to a contract
  * and if the contract is not prepared to refund them they will get stuck in the
contract.
  * The same issue used to happen for Ether too but new Solidity versions added the
payable modifier to
  * prevent unintended Ether transfers. However, there’s no such mechanism for token
transfers.
  * so the below function is created
  */

  function refundTokens(address _token, address _refund, uint256 _value) {
     require (msg.sender == owner);
     require(_token != address(this));
     AbstractToken token = AbstractToken(_token);
     token.transfer(_refund, _value);
     RefundTokens(_token, _refund, _value);
  }

  /**
   * Logged when token transfers were frozen.
   */
  event Freeze ();

  /**
   * Logged when token transfers were unfrozen.
   */
  event Unfreeze ();

  /**
   * when accidentally send other tokens are refunded
   */

  event RefundTokens(address _token, address _refund, uint256 _value);
}
```

3. **Contract ABI:**

```
[{
    "constant": false,
    "inputs": [],
    "name": "freezeTransfers",
    "outputs": [],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": true,
    "inputs": [],
    "name": "name",
    "outputs": [{
      "name": "",
      "type": "string"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "_spender",
      "type": "address"
    }, {
      "name": "_value",
      "type": "uint256"
    }],
    "name": "approve",
    "outputs": [{
      "name": "success",
      "type": "bool"
    }],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "_newOwner",
      "type": "address"
    }],
    "name": "setOwner",
    "outputs": [],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": true,
    "inputs": [],
    "name": "totalSupply",
    "outputs": [{
      "name": "supply",
      "type": "uint256"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "_from",
      "type": "address"
    }, {
      "name": "_to",
      "type": "address"
    }, {
      "name": "_value",
      "type": "uint256"
    }],
    "name": "transferFrom",
    "outputs": [{
      "name": "success",
      "type": "bool"
    }],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": true,
    "inputs": [],
    "name": "decimals",
    "outputs": [{
      "name": "",
      "type": "uint8"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "constant": false,
    "inputs": [],
    "name": "unfreezeTransfers",
    "outputs": [],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "_value",
      "type": "uint256"
    }, {
      "name": "_collected",
      "type": "uint256"
    }],
    "name": "createTokens",
    "outputs": [{
      "name": "success",
      "type": "bool"
    }],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": true,
    "inputs": [{
      "name": "_owner",
      "type": "address"
    }],
    "name": "balanceOf",
    "outputs": [{
      "name": "balance",
      "type": "uint256"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "target",
      "type": "address"
    }, {
      "name": "mintedAmount",
      "type": "uint256"
    }],
    "name": "mintToken",
    "outputs": [{
      "name": "success",
      "type": "bool"
    }],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "_token",
      "type": "address"
    }, {
      "name": "_refund",
      "type": "address"
    }, {
      "name": "_value",
      "type": "uint256"
    }],
    "name": "refundTokens",
    "outputs": [],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": true,
    "inputs": [],
    "name": "symbol",
    "outputs": [{
      "name": "",
      "type": "string"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "constant": false,
    "inputs": [{
      "name": "_to",
      "type": "address"
    }, {
      "name": "_value",
      "type": "uint256"
    }],
    "name": "transfer",
    "outputs": [{
      "name": "success",
      "type": "bool"
    }],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "function"
}, {
    "constant": true,
    "inputs": [{
      "name": "_owner",
      "type": "address"
    }, {
      "name": "_spender",
      "type": "address"
    }],
    "name": "allowance",
    "outputs": [{
      "name": "remaining",
      "type": "uint256"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "constant": true,
    "inputs": [],
    "name": "totalCollected",
    "outputs": [{
      "name": "",
      "type": "uint256"
    }],
    "payable": false,
    "stateMutability": "view",
    "type": "function"
}, {
    "inputs": [],
    "payable": false,
    "stateMutability": "nonpayable",
    "type": "constructor"
}, {
    "anonymous": false,
    "inputs": [],
    "name": "Freeze",
    "type": "event"
}, {
    "anonymous": false,
    "inputs": [],
    "name": "Unfreeze",
    "type": "event"
}, {
    "anonymous": false,
    "inputs": [{
      "indexed": false,
      "name": "_token",
      "type": "address"
    }, {
      "indexed": false,
      "name": "_refund",
      "type": "address"
    }, {
      "indexed": false,
      "name": "_value",
      "type": "uint256"
    }],
    "name": "RefundTokens",
    "type": "event"
}, {
    "anonymous": false,
    "inputs": [{
      "indexed": true,
      "name": "_from",
      "type": "address"
    }, {
      "indexed": true,
      "name": "_to",
      "type": "address"
    }, {
      "indexed": false,
      "name": "_value",
      "type": "uint256"
    }],
    "name": "Transfer",
    "type": "event"
}, {
    "anonymous": false,
    "inputs": [{
      "indexed": true,
      "name": "_owner",
      "type": "address"
    }, {
      "indexed": true,
      "name": "_spender",
      "type": "address"
    }, {
      "indexed": false,
      "name": "_value",
      "type": "uint256"
    }],
    "name": "Approval",
    "type": "event"
}]
```

4. Public functions:



```
freezeTransfers
name
approve
setOwner
totalSupply
transferFrom
decimals
unfreezeTransfers
createTokens
balanceOf
mintToken
refundTokens
symbol
transfer
allowance
totalCollected
```

5. **Token Testing Information:**

   1. **Contract type:** ERC-20 standard Token
   2. **Test environment:** Rinkeby testnet (Ethereum Test platform)
   3. **Contract Address:** 0x3de5885d97828f9c11ef9476307e3b86d25b9e61
   4. **Etherscan link:** https://rinkeby.etherscan.io/token/0x3de5885d97828f9c11ef9476307e3b86d25b9e61
   5. **Token name:** CIBUS Token Test4
      Token Symbol: CBTTEST4
      Decimal points accepted: 10

6. **Test Cases:**
   This section provides testing of different methods with positive and negative scenarios.
   Tx hash has been provided as a provenance , which resides in Rinkeyby testnet Blockchain.
   To check transaction example:
   https://rinkeby.etherscan.io/tx/0x9c2cebfec24f756954627252191639e45c0b39a0bedfdf8cfb7e301e59c748ce

Test cases:

Test case 1 :

Scenario:

Owner access to generate tokens

1. **Create Token** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 (current Owner)

Tx Hash: 0x9c2cebfec24f756954627252191639e45c0b39a0bedfdf8cfb7e301e59c748ce

Result: Success

2. **Mint token** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 (current Owner) to 0x1Bb47c36F34dB4DC246E8AA870A31cf79f829555

Tx Hash: 0x18e2ec513e2d4064c29c8d354001db0b586b2500d73952f391ffe913613aa84e

Result: Success

Test case 2:

Scenario:

Other user wants to create token

3. **Create Token** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0xa837a77c58cf07afda8160ae15fc32d37941956a49a46811bd7a75eee604e86c

Result: Fail

Analysis: This is intended behavior as other user can not create or mint token directly.

4. **Mint Token** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0xb97c1d470234913de5ca71b716eae285dfc636f5266d2482b8af5f6e44cd0ef6

Result: Fail

Analysis: This is intended behavior as other user can not create or mint token directly.

Test case 3:

Scenario:

Change the owner

5. **setOwner** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0x00bc08a551c40a373719e3ce5296f179aa37095dfd2eb6163ecf2d18f647ba4f

Result: Fail

Analysis: This is intended behavior as other user can not change the owner, only current owner can change the ownership.

5. **setOwner** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 (owner)

Tx Hash: 0xd74377a7307bfc9f2c50a820816d35e5b500be5b8f393812e5923aae8a640660

Result: Success

Analysis: This is intended behavior as only current owner can change the ownership.

Test case 4:

Scenario:

New owner can create or mintokens

6. **Create Token** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (current owner)

Tx Hash: 0xe9682afafff05abbf3e50bd8096cc2a71d8887b6af69d38e5d857ff8ff721245

Result: Success

Analysis: This is intended behavior, current owner can create or mint tokens.

Note: Specify high gas value for 20 Gwei gas price specify 71000 gas limit.

7. **Mint Token** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (current owner)

Tx Hash: 0x15ad73296baf519e40bc7cb31e2f1e66e2521fe0078285d15227ca4ab085af20

Result: Success

Analysis: This is intended behavior, current owner can create or mint tokens.

Test Case 5:

Scenario:

Check token balance of an address

8. **Check balance** (CBT Token balance) of account
Read method and working good
No tx Hash is specified as it is read function so block is mined for it.

Test case 6:

Scenario:

Total Supply of tokens at this point of time.

9. **Total Supply** of Tokens.

It will return the number of total tokens created till now in the contract.
Read method and working good
No tx Hash is specified as it is read function so block is mined for it.

Test Case 7:

Scenario:

Token Transfer process

10. **Transfer** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 (current owner)

Tx Hash: 0xa1f6f27e46e4d55933dad1a8582c2e7a66b409accd58a3d16237f5bf007f8173

Result: Success

Analysis: This is intended behavior any token holder can transfer tokens to other account when token freeze flag is false. By default it is false.

Note: gas limit will be 41000 and gas price 21Gwei.

11. **Transfer** in freeze situation from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0x81d7c321ec0cb3b8c63fac3840eb84cc72ccd8c731bb7e48826b319cb280a5e4

Result: Fail

Analysis: This is intended behavior any token holder can not transfer tokens to other account when token freeze flag is true.

12. **Transfer** in unfreeze situation from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0x7b0b3541c3f18bafc90618bc1c6c42e34f483adc318b868f382909d4b7c2866f

Result: Success

Analysis: This is intended behavior, token holder can transfer tokens to other account when token freeze flag is false.

Note: Gas limit will be 41000 and gas price 21Gwei.

Test case 8:

Scenario:

freeze and unfreeze token transfer

13. **freeze transfer** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 (current owner)

Tx Hash: 0x37d792a1e1b5bcfdea7fc3c32b7a7ae5ae01a37ae78f80048e8d340968cabae7

Result: Success

Analysis: Only owner can freeze the token transfer. Token transfer is in freeze state now. By default it was false, now it is true.

Note: Gaslimit must be 51000 and gas price: 21 Gewi

14. **unfreeze transfer** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0x83dd2c5e8aa32c226f121d0cb764b30a997c928cc9ccff07b7ccb60dd1d501e8

Result: Fail

Analysis: Only owner can freeze or unfreeze the token transfer.

15. **freeze transfer** from 0x8ff965527b29ABc895e76eE0E2504816f82F5b95 (not owner)

Tx Hash: 0x0cee995b536c53372feecf26513ab2b45d605af7bc026d1f75aa48b4d3cdf17a

Result: Fail

Analysis: Only owner can freeze or unfreeze the token transfer.

Test case 9:

Scenario:

Approve to send the token on behalf of another token Holder

16. **Approve** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 to 0x8ff965527b29ABc895e76eE0E2504816f82F5b95

Tx Hash: 0x3771a1c46199feec3650b449571e94cddc8977885d81a70ad2aa27814252773e

Result: Success

Analysis: Now delegated account can send the specified amount on behalf of approver token holder.

Note: Gas limit is 51000 and gas price 21Gwei.

17. **Check allowance** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 to 0x8ff965527b29ABc895e76eE0E2504816f82F5b95
Read method and status Success.

18. **Change approval** amount from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 to 0x8ff965527b29ABc895e76eE0E2504816f82F5b95

Tx Hash: 0x281f88970281744e0b84d5e170542f4ab64a8ee8b09c88a620caa5e3d383fcaf

Result: Fail

Analysis: can not change the approval amount unless make it 0 and allocate fresh value. This is to avoid race condition in the code.

Note: Gas limit is 51000 and gas price 21Gwei.

19. **Making 0 and then set approval** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 to 0x8ff965527b29ABc895e76eE0E2504816f82F5b95

Tx Hash: 0xba356932aba4a0dc6594b2cb6d3af5e73abc3c63ddb54d2ca6624b6277f0b25a (to make 0)
and 0x82d1223b85e4603b908854c939eddda7620290b58b1c74c576cf904a4bb55af1 (to make new approval amount)

Result: Success

Analysis: Now delegated account can send the specified amount on behalf of approver token holder.

Note: Gas limit is 51000 and gas price 21Gwei.

Test case 10:

Scenario:

Transfer from 'part' or 'full' allowance amount to a different account.

20. **Transfer from** 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 to 0x1Bb47c36F34dB4DC246E8AA870A31cf79f829555 by 0x8ff965527b29ABc895e76eE0E2504816f82F5b95

Tx Hash: 0xdd07a85acf09773a4783a26584e4baf981f0e2aab83b86680727b5d8c592acc4

Result: Success

Analysis: it is intended behavior as the amount can be examined allowance function.

Note: Gas limit is 51000 and gas price 21Gwei.

20. **Transfer excess than delegated amount from** 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2
to
0x1Bb47c36F34dB4DC246E8AA870A31cf79f829555
by
0x8ff965527b29ABc895e76eE0E2504816f82F5b95

Tx Hash: 0xa95d7c5b6cca1dab1edf1080fbed21792e3ac5fec89825d68cd7b33ea6ac3803

Result: Fail

Analysis: More than delegated amount cannot be transferred

Note: Gas limit is 51000 and gas price 21Gwei.

Test case 11 :

Scenario:

Accidentally send other tokens locked into the contract, so refund Tokens function is used to send back to the user.

21. **Refund Tokens** from 0xeA4387E32e312C1D5141565f59c9F514f3C08ca2 via the contract.

Tx Hash: 0x538c0ed99f1ce57d2960fe362b6ed4ac4e6a270429a7915f4cf646d0c456b043

Result: Success

Analysis: Different token called CIBUS TOKEN TEST2 was transferred to the contract. which was now in the

contract address, now owner sends back the other type token to accidentally send token holder's account.

Note: Gas limit is 61000 and gas price 21Gwei.