International Telecommunication Union

DIGITAL CURRENCY GLOBAL INITIATIVE

TELECOMMUNICATION
STANDARDIZATION SECTOR

(03/2023)

Digital Currency Ontology Report

Architecture, Interoperability Requirements and Use Cases (AIRU) Working Group

DISCLAIMER

The findings, interpretations, and conclusions expressed in this work do not necessarily reflect the views of Digital Currency Global Initiative partners, including the International Telecommunication Union, or Stanford University. The mention of specific companies, or of certain manufacturers' products does not imply that they are endorsed nor recommended by ITU in preference to others of a similar nature that are not mentioned. Errors and omissions excepted; the names of proprietary products are distinguished by initial capital letters. The Digital Currency Global Initiative partners do not guarantee the accuracy of the data included in this work. The boundaries, colours, denominations, and other information shown on any map in this work do not imply any judgment on the part of the Digital Currency Global Initiative partners concerning the legal status of any country, territory, city or area or of its authorities or the endorsement or acceptance of such boundaries.

Table of Contents

Table of Figures

1 Ontology as Anchor Information Construct

This report aims to demonstrate the critical importance, utility, and distinctions of an ontology as the construct of "information organization" in defining the "thing we are talking about," as compared to other more traditional organizational structures such as a taxonomy, classification, framework, or tabulation. As such, an ontological "method-of-development" is also described with associated rules and constraints to ensure the outcome of the more stringent ontological method successfully produces the unique characteristics of an ontology. It also ensures that the process does not easily and naturally degenerate into lower form and therefore lower utility constructs, such as frameworks which are challenged by issues of completeness and correctness.

The absence of a common understanding of a reliable representation of "the thing we are talking about," before requirements are defined and mandated for it, yields only notional requirements that do not deliver the assurance levels necessary for protection of complex systems in the digital age.

Putting aside for a moment, "the thing we are talking about," what does an ontology versus a taxonomy, classification, framework, table or list have in terms of its ability to reliable capture and retain knowledge about the "thing," thereby enabling a reliable multi-disciplinary and integrated analysis of the thing we are all concerned about protecting?

- In philosophy, **ontology** is the branch of metaphysics that studies the nature of existence or being. In computer science, an ontology is a **formal** and **explicit specification** of a **conceptualization** of a **domain** of knowledge. In other words, an ontology is a way of **representing knowledge** about a particular domain, such as a field of study or an industry, in a structured way that **allows for precise and unambiguous communication between different elements of a system through their relationships**. It typically consists of a vocabulary of terms, along with rules for **combining** those terms to create **more complex concepts** and relationships between them. Ontologies are used in many applications, including artificial intelligence, natural language processing, and the semantic web. They allow computers to understand the meaning of information and to reason about it **in a more intelligent way**, which can lead to less complex, more efficient, and effective communication, analysis, and decision-making.
- A **taxonomy** is a way of **classifying** and **categorizing** things based on their **characteristics** and **relationships**. It is often used in biology to classify living organisms into different groups based on their physical and genetic characteristics. In information science and knowledge management, a taxonomy is a way of organizing information or knowledge in a **hierarchical** structure, where each level of the hierarchy represents a **more specific or detailed** category of information. This allows for more efficient and effective retrieval and organization of information. Taxonomies can be used in many different applications, such as e-commerce websites, content management systems, and search engines. Overall, a taxonomy is a tool for organizing and categorizing information, which can be used to make --information **more easily accessible** and understandable.
- **Classification** is the process of **organizing** or **categorizing** items or concepts into groups based on their similarities or differences. It is a fundamental process that helps us make sense of the world by identifying and grouping similar things together. Classification can be based on many different criteria, such as physical characteristics, function, behavior, or purpose. For example, in biology, organisms are classified based on their physical characteristics and genetic makeup. In library science, books are classified based on their subject matter and content. In data science, classification is a type of machine learning technique that involves training a model to recognize patterns in data and classify new data based on those patterns. The goal of classification is to make it easier to understand and organize complex information, and to enable more efficient and effective processing of that information. It is a powerful tool that is

used in many different fields and applications, from science and technology to business and everyday life.

- A **framework** is a set of rules, guidelines, or models that provide a structure for organizing and understanding a complex system or concept. It is essentially a set of pre-established assumptions, concepts, and practices that help to guide and inform a particular area of inquiry or practice. Frameworks are used in a wide variety of contexts, including software development, project management, education, and research. For example, in software development, a framework might provide a set of standard libraries, design patterns, and coding conventions that help developers build applications more efficiently and with fewer errors. In project management, a framework might provide a set of best practices for planning, executing, and controlling projects. Frameworks are often developed through collaboration between experts in a particular field, and they can help to establish common terminology, best practices, and standards. They can also provide a way to communicate complex ideas and concepts in a clear and concise manner. Overall, a framework is a valuable tool for helping to structure and organize complex information, and for providing a common language and set of practices within a particular field or community.
- In computer science, a **table** is a collection of data organized into rows and columns. Tables are used to store and represent structured data in a way that is easy to read and analyze. They are a common data structure used in relational databases, spreadsheets, and other applications. A table is typically composed of one or more columns, each of which represents a particular attribute or type of data, such as a name, date, or numerical value. Each row in the table represents a specific instance of the data and contains values for each of the attributes defined in the columns. Tables are often used for tasks such as storing and retrieving data, performing calculations and analyses, and displaying information in a structured format. They can be manipulated using specialized software tools or programming languages, which provide a variety of operations for managing and working with table data. Overall, tables are a powerful tool for organizing and working with structured data in a variety of applications, and they are a fundamental concept in many areas of computer science and information --technology.
- In computer science, a **list** is a collection of items that are ordered and can be accessed by their position or index. Lists are used to store and represent sequences of data, such as a series of numbers, names, or objects. A list can be created and modified dynamically, meaning that items can be added, removed, or rearranged as needed. Lists can also be used to perform operations on their contents, such as sorting, searching, or filtering. In programming, lists are a fundamental data structure, and are often used to represent arrays or vectors. They are typically implemented as an array or linked list and can be accessed using various programming constructs such as loops, iterators, or list comprehensions. Lists are widely used in many different types of applications, from web development and data science to gaming and mobile apps. They provide a flexible and efficient way to manage and process data and are --a valuable tool for working with sequences of information in a variety of contexts.

Ontology, taxonomy, classification, framework, table, and list are all terms that are used in different contexts to represent different concepts. However, there are some similarities and differences between these concepts. All these terms are used to represent structures or systems for organizing information. They all involve the use of categories or groupings to organize and classify data. They all facilitate the understanding and processing of complexity.

Ontology on the other hand is a formal and explicit representation of concepts and their relationships in a particular domain, while taxonomy is a hierarchical structure that organizes categories based on their characteristics. Classification is the process of assigning items or concepts to specific categories, while taxonomy is the specific hierarchical structure that defines those categories.

A framework is a set of rules, guidelines, or models that provide a structure for organizing and understanding a complex system or concept, while a taxonomies or ontologies are more specific structures for organizing information. A table is a collection of data organized into rows and columns, while a list is a collection of items that are ordered and can be accessed by their position or index. Tables are often used for storing and organizing data in a structured format, while lists are often used for representing sequences of data.

Taxonomies, ontologies, or classifications can be implemented using tables or lists, but the former three are higher-level concepts that define the organization of the information, while the latter two are lower-level data structures used to store the information.

2 Ontology Development Rules

With an understanding of what an ontology is, compared to other data organization structures, the following will define the assumptions, rules, and criteria by which an ontology is developed.

One: There can only be one (1) ontology representing a given domain scope.

Completeness: Generally, frameworks contain requirements that were defined without any knowledge of the thing it is to be applied to. Consequently, there is no guarantee of completeness. That is, the coverage of the requirements in frameworks are insufficient for what is necessary to fully cover the thing.

- **Bounded:** An ontology must be of a well bounded, specific, and explicit scope of knowledge or of a domain. The ontology must, by definition, cover "all matters" within the defined scope. For example, "all matters" Cryptographic Processes that cover all keyed and non-keyed cryptographic operations for encryption, digital signatures, and authentication.
- **Decomposition**: Once the breadth of the scope is defined, a principle of completeness must apply in the decomposition process of defining, with increasing precision, the structure within the scope. Referred to as de-compositional Completeness, any topic ("parent") can be broken down into two or more sub-topics ("children"). The sum of the scopes covered by each sub-topic (children) must equal the original topic (parent) Scope. This rule ensures no loss of scope in the decomposition process.

Correctness: generally, there are many frameworks that describe differently the same set of requirements to be applied to the same components, for example, the application of encryption while information is stored. Which one is correct?

- "The thing we are talking about" will be ultimately described in words of a particular language. Whether a particular word in a statement is correct versus another is subjective and a matter of linguistic debate. Consequently, the goal is not to "select a word that all can agree with," but to drive towards "the natural correctness of a word" given its context by adopting a few "choice of words" principles. The objective is to minimize subjectiveness and maximize clarity.
- The first is related to the choice of words within a language. Some words, such as "jargon," are to be avoided because they are context specific and come with preconceived past notions or "baggage." On the other extreme, neutral, agnostic technical terms are clear and concise and context independent. For example, the choice of words to describe where digital currency is stored. A "wallet" is often used as it is where we place physical currency. Selecting jargon is often a choice made to bring forward the past meaning into the future. The problem is that jargon is language specific and jargon in one language does not remain consistent in another

language. On the other hand, this ontology uses “digital currency store” as a technology neutral term that means only one thing – where digital currency is stored.

3 Building Blocks and their Relationship

Elemental Building Blocks

A core aspect of building an ontological model is the concept of defining “elemental” building blocks at the lowest and simplest level possible. Given the inherent narrow scope of a building blocks, their correctness can be demonstrated more easily and with higher confidence.

Building Block Assembly into more Complex Constructs

More complex constructs are created by “assembling” simpler building blocks. That is, more complex constructs are not defined independently of other simpler constructs but must be defined by the simpler ones. This constraint increases dramatically the consistency of elements and their relationships. By extension, the integration of two “correct” building blocks, through a logical relationship, results on a larger more complex building block that can be demonstrated to be also correct.

Ontological Terminology: Ontologies have key architecture terms referred to as “notions”, “distinctions”, and “values”. The following will define these terms and their relationships.

- An ontology is first described by of a set of high-level notions, referred to as “level 1 notions”. Each notion is separated from others at the same level by a fundamental distinction present and unique to the “thing being modelled.”
- Notions must be **mutually exclusive** from, each other. There cannot be any coverage overlap. Aspects of one notion cannot exists in another.
- Each level 1 notion is then in turn sub-divided into distinctions, fundamental differences in that notion that must be unique: described, accounted for, and located once, and nowhere else in the ontology.
- The cumulative coverage of a notion’s distinctions must equal that of the notion. That is, distinctions of a notion must be complete.
- Level 1 notions decompose into level 2 distinctions; these level 2 distinctions become level 2 Notions which in turn can decompose further into level 3 distinctions. This process continues for each individual notion separately until no further distinctions can be made and only values of the distinctions can be provided. This is the “bottom” of one path of the distinction tree.

Distinction Interdependence:

As was mentioned previously, the building block and assembly method central to building an ontology, yields a highly interdependent information construct. Selecting values of one distinction, impacts, constrains, or determines other distinctions and their values.

One Ontology contains all Derivatives:

The implication of an ontology is that only one can exist for a given scope, so that all derivatives (forms) can be generated by the same ontology simply be selecting different distinction values. Change the value of one distinction and the outcome is a different instantiation. For example, the “digital currency supply” ontology must cover all digital currency types. One type is defined by all distinctions having been assigned a value. Change one value of one distinction, and you have a different type.

4 Digital Currency Supply | Agree | Market Ontology

This report describes the outcome of an ontological development initiative to model three independent yet interrelated ontologies. The three ontological models are:

- **DCT supply ontology** defining how digital currency types (DCTs) are created and supplied.
- **DCT agree ontology** defining how DCTs are involved in agreements involving other DCTs and other digital assets (DA);
- **Digital market ontology** (DMO) defining how agreements are executed in a market of multiple DCTs and DA Types

The ontology development, refinement, and enhancement processes are conducted in a software application with a many-to-many relational database providing the ability to capture unique relationships, and therefore knowledge between ontology notions and distinctions. The flattened current status of this work is contained in Section 5.

The model makes the following assumptions:

- **Scope**: all digital currency types
- **Value**: Determined by a market driver and expressed in the unit value of a DCT unit.
- **Ownership**: The legal right to control value and to transfer and transact with it.

5 Background and prior ITU-DCGI work

The model discussed is based on, and supersedes, prior ITU-DCGI S&A work entitled "Digital Currency Conservation of Supply Model," published by Jacques Francoeur, Team Lead of Security WG (Working Group), illustrated in Figure 1.

The model divided the universe of all DCs according to one separation rule. On the left, changes in available DC supply can occur, and on the right, changes in available supply cannot occur. On the right, a Move involves subtracting DC Amount of value from one Digital Currency Store and to deposit the same amount in a destination store. The left side of the model allows the supply of DC to be issued centrally, de-centrally, or distributed while all moves on the right side of the model occur between a sender Store and a receiver Store on a direct peer-to-peer network.

All Moves occur according to the following rule.

Move = Subtract "-" Origin Amount from Source Store
"- condition -"
Add "+" Same Amount to Destination Store

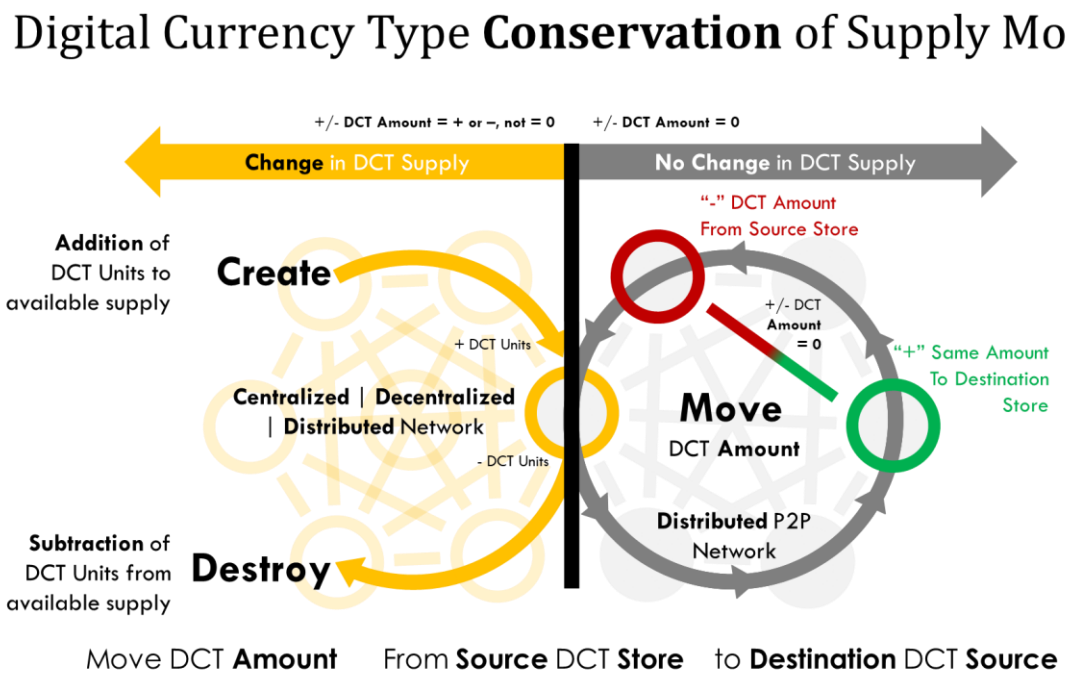


Figure 1: Prior Superseded Model - Digital Currency Type Conservation of Supply Model

6 Single digital currency type ecosystem model

The previously published model is replaced by the following 2nd generation model, greatly enhanced by a parallel ontology development process.

Each DC Type ("Type") creates a single DC Type Ecosystem ("Ecosystem") where the Type exists, is maintained, and engaged in value transfer within the same Type, referred to as Intra-Ecosystem.

The following will describe the formation of the Ecosystem model in stages defined by a **Genesis Time Sequence**, a series of related and interdependent time increments relative to Time = 0, defined as when the Ecosystem is available for its 1st unit swap, referred to as the **Genesis Transfer**. Components of the model that are laid out in a time-sequence that respect certain ontology rules about what exists and does not at any point in time.

The top pinnacle notion of this Ecosystem model is Owners and non-Owners of Units and therefore Value. Following this notion, the key distinctions of "Access is Ownership." Consequently, the Owner has the legal right to exercise any action on that Value.

The overview of Genesis Time Sequence can be described as follow:

- @ Time = **-2**: Issuance Digital Store Availability & Genesis Unit **Production**
  By this time, the production of the number of Genesis Units has occurred and are maintained in the Genesis Digital Store.
- @ Time = **-1**: **Move** Digital Store **Availability** & Non-Owner Participant **Onboarding**
  By this time, the finite types of non-owner Participants, referred to as Entrant Participants ("Entrants") have been onboarded and are in control of a Store which contains no Units.
- @ Time = **0**: Creation of **Genesis Owner** Participant(s)
  By this time, the Issuer (configuration of Issuance Participants) executes the 1st transfer of Units at an agreed Unit Value. Referred to as the Genesis Transfer, it determines the first instance where the Issuer Initial Value Offer "crosses" a Participant Bid.
- @ Time = **+1**: **Open** & **Active** Single Digital Currency Ecosystem
  By this time, a supply of Units is in circulation among Ecosystem Participants and the Ecosystem is open for business.

6.1 Time = 0-2: Issuer store availability and genesis unit production

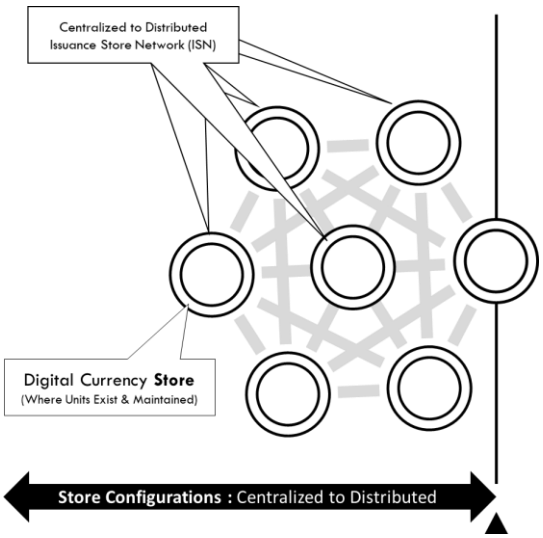


Figure 2: Issuance Configuration Store Availability

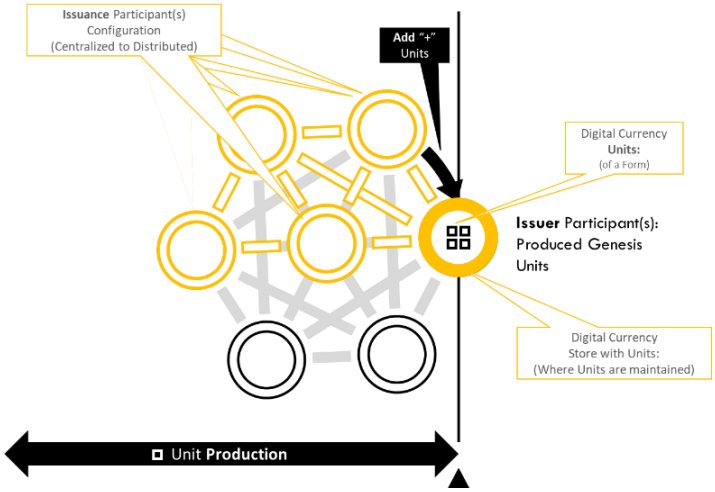


Figure 3: Production of Genesis Units

As illustrated in Figure 2, @ Time = -2^{-}, all Digital Stores involved in the Unit Production Process, referred to as the Issuance Configuration ("Configuration"), are available, a precondition of proceeding forward. A Digital Currency Store (DCS, "Store") is "where Digital Currency Units ("Unit") exists, are maintained, and updated. The Configuration can be centralized, decentralized, or distributed. Configuration determines the Digital Currency Type ("Type"), in part. For example, a Digital Currency issued centrally by a central bank yields a Central Bank Digital Currency, issued centrally by a private bank yields a Stablecoin; while issued decent rally yields a cryptocurrency.

As illustrated in Figure 3, @ Time = -2^{+}, the Issuance Configuration executes the Genesis Protocol to produce the number of Genesis Units which are stored and maintained in the Issuer Genesis Store.

A Digital Currency Type (DCT) (“DC”) is a digital representation of Value. A Digital Currency Unit (DCU, “Unit”) has a Digital Currency Form (DCF, “Form”) which is defined by the digital data architecture/structure of a Unit – how it is built.

A Digital Currency has a “State,” even with its availability, the usability of the Unit at any given time can be: **Inactive** (e.g., pre-value Unit, Hold (external control); **Locked** (Internal programable event), or **Active** State (post-value event).

6.2 Time = 0-1: Participant store availability and onboarding

In the same way an Issuance Configuration of Issuer Participants with Stores are required to produce Units, @ Time = -1-, a peer-to-peer network of Participant Stores is required to onboard Participant Types ("Participants"), illustrated in Figure 4.

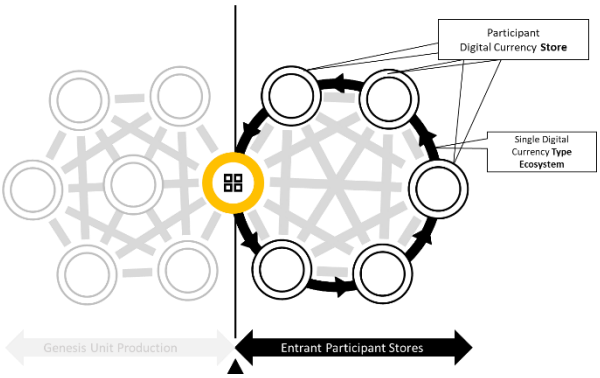


Figure 4: Availability of Participant Stores

At this point, the Issuer Participant(s) exists, and the Genesis Units are produced and maintained in the Genesis Store. Next, non-Issuer Participants of specific types will onboard into the DCT Ecosystem. These Participant Types have a specialized and specific **function** or purpose.

All new Ecosystem **Participants** are 1st an **Entrant** Participant ("Entrant") and will become a specialized. All **Entrants** are **non-Owners**, a binary State-of-a-Participant: **Owner/Non-Owner.** Entrants can remain non-Owners, become Specialized Participants, or become Owners. Participants can either be **Inactive** or **Active.**

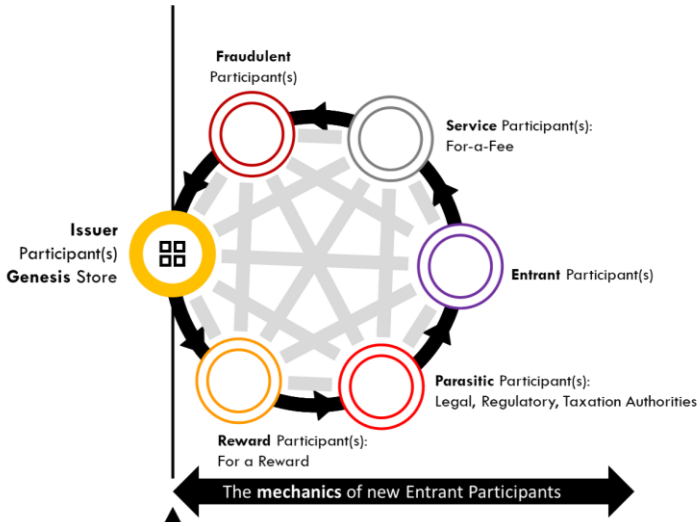


Figure 5: Entrant Participant Onboarding

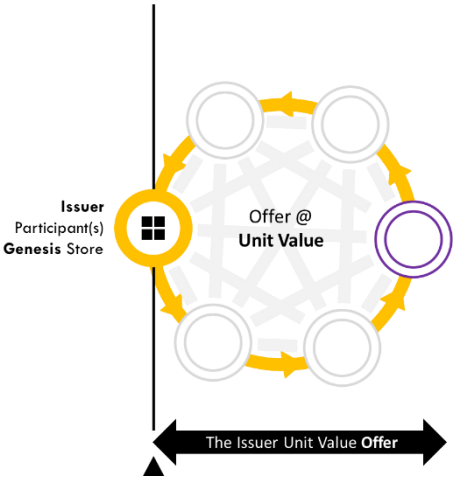
Entrant Types:

- **Service** Participant**:** Entrant become a Service Participant ("Service Provider") performing a Service for-a-Fee in an Agree.
    - **Synchronous** Multiple Service Provider **Pass-it-Forward** Fee Model: More than one Service Providers can provide Services in series when the Buyer commits the Seller Amount plus all Service Fees. Each Service Provider extracts their Fee and passes the **Remainder Forward** to the next Service Provider.
    - **Asynchronous** Multiple Service Provider: payment of fees occurs after (**post-Agree**) or before (**pre-Agree**)
- **Reward** Participant**:** Entrant become a **Reward** Participant ("Validator, Verifier") performing a function for remuneration in an Agree.
- **Fraudulent** Participant: a legitimate Participant under the control of a malicious actor.
- **Parasitic** Participants: Legal, Regulatory, and Taxation authorities that impose "friction" and costs to the DCT Ecosystem.

@ Time = **-1**$^{+}$: Entrant Participant of all types begin **Onboarding** which is an ongoing process.

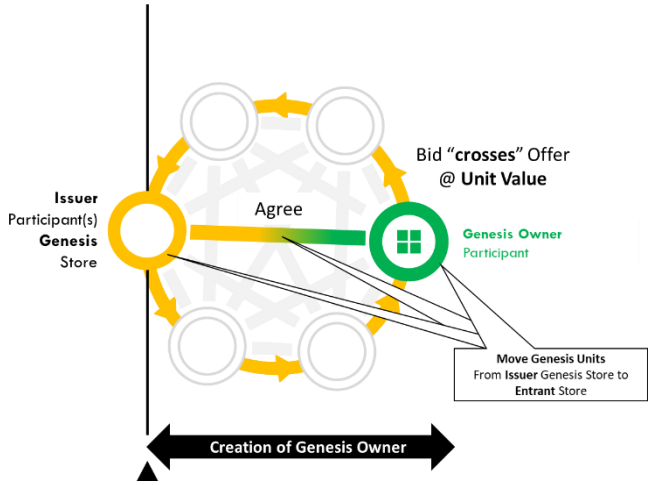
6.3 Time = 0: creation of genesis owner – first value move, unit value

By this time, **Genesis Units** exist and are maintained in **Genesis Store**. Non-Owner Participants including Service and Reward Participants; Parasitic Participants have been onboarded. Now is the time to create the 1st Owner Participant, referred to as they **Genesis Event**.



One or more Entrants become the first "owner" Participant(s), referred to as the **Genesis Participants**. The following is the process involved in the **Genesis Transfer**, the formation and execution of the **Genesis Agree**, the only first event of its kind.

The critical outcome of creating the Genesis Owner is the establishment of the **Unit Value**. The Issuer makes an Offer @ a specific Unit Value, referred to as the **Initial Value Offer** (IVO). Entrant(s) submit Bids. When Entrant Bid(s)s cross IVO, the Genesis Agree "**Cross**" condition is met. Issuer moves **Genesis Amount** (Unit Value x # of Units) from Genesis Store to Genesis Participant(s) Stores. More specifically,



- Issuer Commits Genesis Units (locked) to be transferred to each Genesis Participant.
- Genesis Amount is Removed/Subtracted "-" from Genesis Store
- Genesis Amount is Transferred & Deposited/Added "+" to one or more Genesis Participant Stores
- Ecosystem Supply = # of Units in Genesis Issuance

6.4 Time = 0+1: Open single digital currency ecosystem

As illustrated in Figure 6, the Issuance Store Network or Issuance Configuration Network of Digital Stores is combined at the Issuer Digital Store junction with the Participant Store Network.

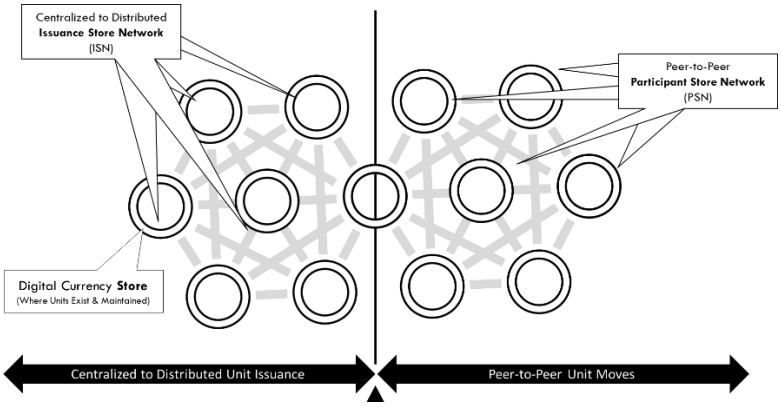


Figure 6: Issuance Configuration Store Network linked to Move Store Network

As illustrated in Figure 7, once all the Participant Types are active and the **Genesis Event** has occurred, a single Digital Currency Type Ecosystem Model is created.

@ Time = +1: **Open** Single Digital Currency Type Ecosystem (DCE, "Ecosystem").

Ecosystem (all locations) where DC is created (+Unit), stored, maintained, and otherwise exists; and transferred, transformed, processed, and destroyed (-Unit).

@ Time = $+1^{+}$: **Active** Single Digital Currency Type Ecosystem. By this time, a supply of Units is in circulation among Ecosystem Participants and the Ecosystem is open for business and actively transferring value.

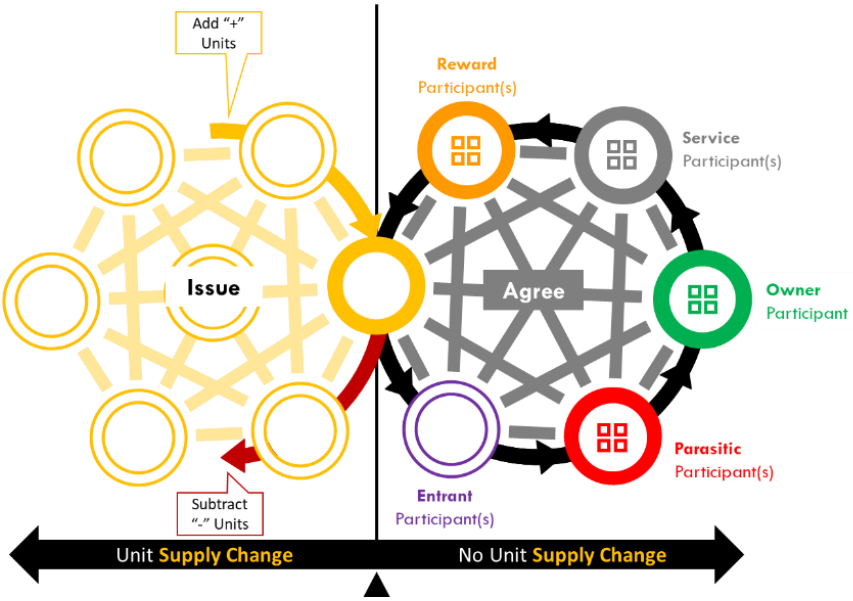


Figure 7: Single Digital Currency Type Ecosystem with all Participant Types

7 Inter-digital currency type ecosystem model

7.1 Inter-digital currency form model

As illustrated in Figure 8, the Single Digital Currency Ecosystem A can be replicated to represent a second Digital Currency Form B Ecosystem. As previously discussed, the two ecosystems can conduct intra-agreements within their own Ecosystems. The Model illustrated in Figure 9 expands to represent the interaction between 2 different Digital Currency Form Ecosystems.

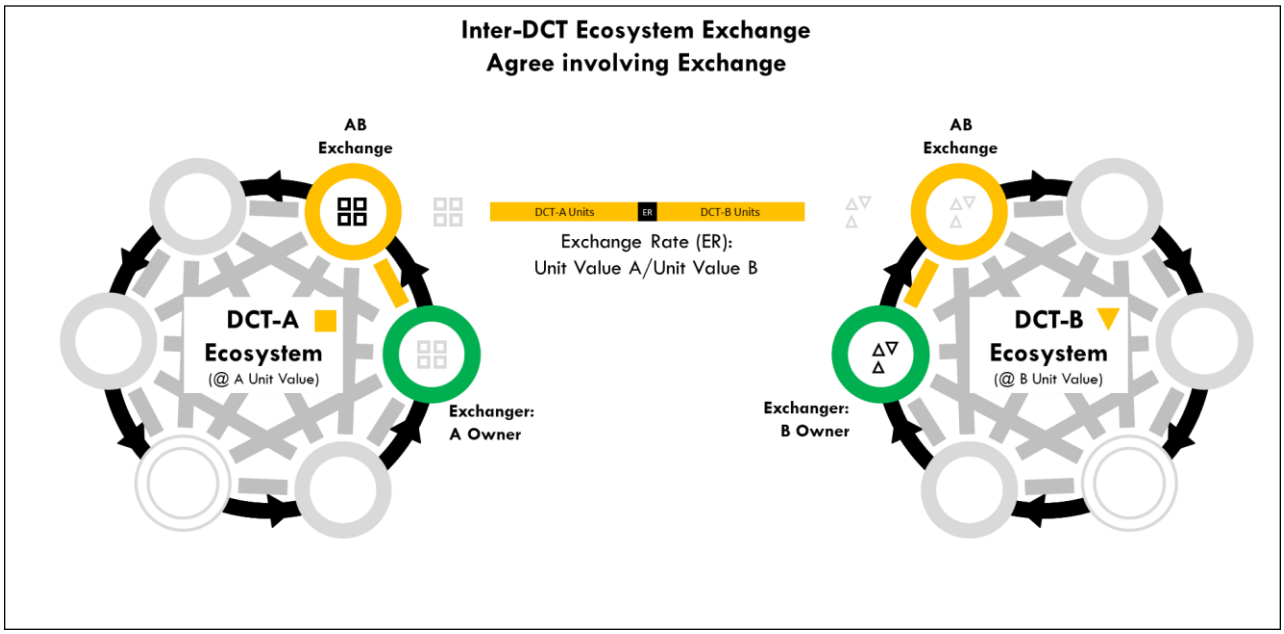


Figure 8: Inter-Ecosystem Agree Exchange of DCT-A for DCT-B

More information on this model can be found at the technical report “Enabling KYC and AML verification on User Transactions” (S. Moreno & Seigneur, 2022).

8 Digital market ecosystem ontologies: Supply, agree, market

The DC world is diverse in terms of form, function, architecture, and uses. For example, issuance can be decentralized in the case of cryptocurrencies (CC) and centralized in the case of commercial bank demand accounts, with other forms in between. The degree-of-decentralization is not the only key characteristic of a DC. What are the others? What makes one DC different from another? What are the core elements of all DCs? How does DC move value around? How can value be moved between DC Forms. How are services paid for, rewards granted, ownership acquired and transferred.

These questions, and many more that were discovered, have evolved the scope of work to define a full DC Market Ecosystem Ontology. A framework of notions composed of distinctions, each composed of sub-distinctions that are architypes, “DNA-like” reference architecture components that must be agnostic and complete in what they address.

The persistent application of ontological and de-compositional techniques in the development of this work consolidated a large set of notions into three fundamental 1st-level ontology notion groups: DC Supply, DC Agree and Agree Market.

1. **DC Supply Ontology** covers the notions and distinctions related to representing all possible DC forms, while being able to define anyone specifically.
2. **DC Agree Ontology** covers the notions and distinctions related to an agreement between two participants involving the movement of value.
3. **Agree Market Ontology** relates to the market where all DC Agrees occur between any and all DC Forms.

As the larger set of notions consolidated into three, notions became sub-notions falling under one of the three. The consolidation, however, must maintain the connection and interaction between DC Supply injected into one Agree Market to conduct DC Agrees.

8.1 Supply ontology

The following provides the definitions for various terms related to digital currency, including the currency itself (Digital Currency or "DC"), the units that make up the currency (DC Unit or "Unit"), the overall supply of units (DC Unit Supply or "Supply"), and the different states that units can be in (Unit State, Inactive Unit State, Active Unit State, Locked Unit State). It also defines terms related to the data architecture of the currency (DC Form, DC Store, DC Type), the ecosystem in which the currency exists (DC Ecosystem), and the ways in which the supply of units can change (Unit Supply Change, Fixed Unit Supply, Variable Unit Supply, Increase/Decrease Unit Supply, Unit Change by Issuer Policy, Unit Change by Issuer Algorithmic, Unit Change by Issuer Oracle, Unit Change by Issuer Voting, Unit Supply Release Schedule, Unit Supply Distribution). It also differentiates between physical and non-physical units, and between different types of data structures used to represent the currency. Additionally, it defines terms related to unit Form Properties such as Fungibility, Rights and Acceptance.

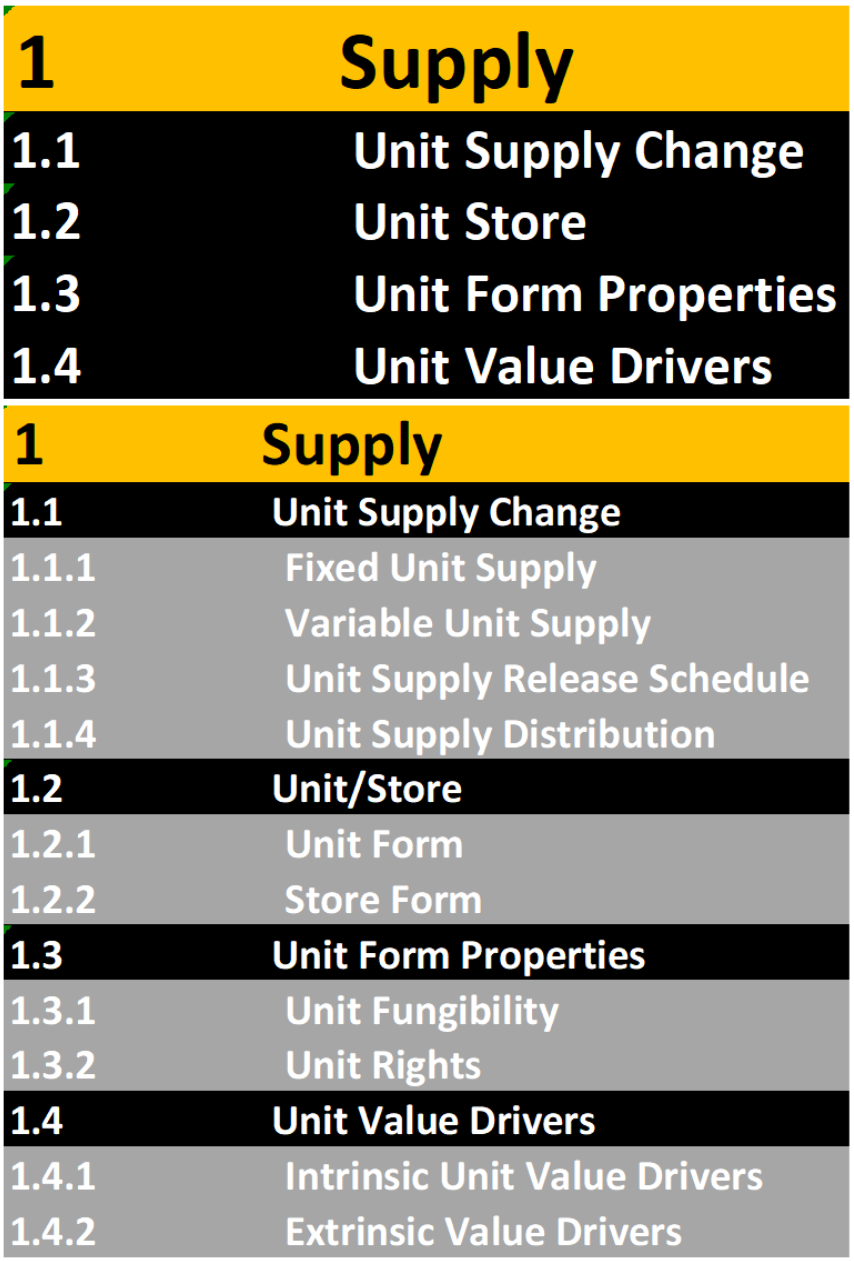


Figure 9: Digital Currency Supply Ontology: Level 1 and 2

1	**Supply**
1.1	**Unit Supply Change**
1.1.1	Fixed Unit Supply
1.1.2	Variable Unit Supply
1.1.2.1	Increase in Unit Supply
1.1.2.2	Decrease in Unit Supply
1.1.3	Unit Supply Release Schedule
1.1.4	Unit Supply Distribution
1.2	**Unit/Store**
1.2.1	Unit Form
1.2.2	Store Form
1.3	**Unit Form Properties**
1.3.1	Unit Fungibility
1.3.2	Unit Rights
1.3.2.1	Native Unit Rights
1.3.2.2	Programed Unit Rights
1.3.2.3	Assigned Unit Rights
1.4	**Unit Value Drivers**
1.4.1	Intrinsic Unit Value Drivers
1.4.1.1	Driven by Supply & Demand
1.4.1.2	Driven by Tokenization of Intrinsic Value
1.4.2	Extrinsic Value Drivers
1.4.2.1	Guaranteed by Entity
1.4.1.2	Guaranteed by Issuer
1.4.1.3	Guaranteed by Backer
1.4.2.4	Backing Collateral Requirement
1.4.2.5	Backing Collateral Type
1.4.2.6	External Pegging Mechanism

Figure 10: Digital Currency Supply Ontology: Level 3

1	Supply
1.1	Unit Supply Change
1.1.1	Fixed Unit Supply
1.1.2	Variable Unit Supply
1.1.2.1	Increase in Unit Supply
1.1.2.2	Decrease in Unit Supply
1.1.3	Unit Supply Release Schedule
1.1.4	Unit Supply Distribution
1.2	Unit/Store
1.2.1	Unit Form
1.2.2	Store Form
1.3	Unit Form Properties
1.3.1	Unit Fungibility
1.3.2	Unit Rights
1.3.2.1	Native Unit Rights
1.3.2.1.1	Unit Acceptance
1.3.2.1.2	Unit Voting Rights
1.3.2.1.2.1	Vote Accessibility
1.3.2.1.2.2	Voting Eligibility
1.3.2.2	Programed Unit Rights
1.3.2.2.1	Controlled Purposes
1.3.2.2.1.1	Use Constrained
1.3.2.2.1.2	Denomination Adjustable
1.3.2.2.2	Single Function Non-Programmable
1.3.2.3	Assigned Unit Rights
1.3.2.3.1	Assigned Legal Rights
1.4	Unit Value Drivers
1.4.1	Intrinsic Unit Value Drivers
1.4.1.2.1	Fungible Tokens
1.4.1.2.2	Non-Fungible Tokens
1.4.2	Extrinsic Value Drivers
1.4.2.1	Guaranteed by Entity
1.4.1.1	Guaranteed by Issuer
1.4.1.1.1	Public Sector
1.4.1.1.2	Private Sector
1.4.1.2	Guaranteed by Backer
1.4.1.2.1	Backer Degree-of-Separation
1.4.2.2	Backing Collateral Requirement
1.4.2.3	Backing Collateral Type
1.4.2.4	External Pegging Mechanism

Figure 11: Digital Currency Supply Ontology: Level 5

The following are the Supply-related definitions.

1. **Digital Currency** ("**DC**"): a representation of monetary Value in digital form.
2. DC **Unit** ("**Unit**"): a DC must exist 1st at time of Issuance as a singular Data Structure that is produced whenever Supply increases.
3. DC **Uni**t Supply ("**Supply**"): the number of Units available as Supply in the Single Digital Currency Type Ecosystem ("Ecosystem").
4. Unit **State ("State")**: Units from their time of production to their removal from Supply, even during their availability, their **usability** can be controlled as follows:
5. **Inactive** Unit State ("**Inactive"**): non-operational state of the Unit after production and before Unit Value Determination and during removal of Units from Supply.
6. **Active** Unit State ("**Active"**): normal operational state of the Unit available in Supply.
7. **Locked** Unit State: as part of the Unit's normal operating state, a Unit can be locked and functionally unavailable. For example, in the cased of a programable Type, the Unit can enter a hold period.
8. DC **Form** ("**Form**"): The digital data architecture/structure of a Unit – how it is built.
9. DC **Store** (**"Store"**): The digital data architecture of where the Form exists and is maintained.
10. DC **Type** ("**Type**"): is defined by the combined attributes of Unit and Store.
11. DC **Ecosystem** ("**Ecosystem**"): means the ecosystem (all locations) where DC Type is created (+Unit), stored, maintained, and otherwise exists; and transferred, transformed, processed and destroyed (-Unit).
12. **Mechanical** Ecosystem: means the Ecosystem operational state independent of unit value determination, Unit state: Inactive
13. **Unit Supply Change**: instances where the # of Units available in Supply may vary.
14. **Fixed** Unit Supply: No change in the # of Units available in Supply
15. **Variable** Unit Supply: Changes, either increase or decrease, in the # of Units available in the DC Type Ecosystem.
16. **Increase/Decrease Unit Supply**: Increase in the # of Units available in Supply.
17. **Unit Change by Issuer Policy**: Increase/Decrease in Units determined by Policy.
18. **Unit Change by Issuer Algorithmic**: Increase/Decrease in Units determined by algorithm.
19. **Unit Change by Issuer Oracle**: Increase/Decrease in Units determined by Oracle, an authentic and objective source of empirical (non-subjective) data
20. **Unit Change by Issuer Voting**: Increase/Decrease in Units determined by a Voting Process
21. **Unit Supply Release Schedule**: the release of the # of Units produced can be all at once (immediate); over time (Gradual); or based on meeting one or more conditions (conditional).
22. **Unit Supply Distribution**: Issuer will move the Units from Issuer Store (Origin Store) to their destination Store, either directly (e.g., Genesis Units) or through an intermediary (Indirect)
23. **Physical Unit**: Currency that exists in "hard" form.
24. **Non-Physical Unit**: Currency that exists in "soft" form, electronic or digital form. An Electronic Currency and a Digital Currency
25. **Entry Data Structure**: A DC Form that is based on an entry in a data cell that is maintained in a database.
26. **Object Data Structure**: A DC Form that is based on a cryptographic object that is maintained in a File Repository
27. **DC Store Form** refers to whether the Unit in the Store can self-verify its integrity or not.
28. **Store Self-Verification Incapable** means integrity verification is dependent on the Store itself to determine and maintain integrity.
29. **Store Self-Verification Capable** means integrity verification is independent of the Store itself and can be determined through intrinsic self-verification.

30. **Unit Form Properties** are used to describe the characteristics of the Unit including Fungibility, Rights, and Acceptance.
31. **Unit Fungibility**, defined as either fully fungible, where the Unit can be converted into same Value in other Forms and denominations or not.
32. **Unit Rights** are defined as either Inherent, Inherited or Assigned Rights,
33. Unit **Inherent** Rights, that is intrinsic to the Unit itself such as right of Ownership,
34. Unit **Inherited** Rights, that is rights granted due to an executed agreement; or
35. Unit **Legal** Rights: rights granted to the Unit by the national government.
36. Unit **Acceptance** Rights, that is whether the Units must be accepted as payment or mandated; mandated with some exceptions; or under the discretion of the Receiving Party, voluntary basis.
37. **Unit Voting Rights** which is based on the Unit Vote Accessibility and Unit Vote Eligibility
38. **Unit Vote Accessibility**, which can be restricted and unrestricted.
39. **Unit Vote Eligibility**, which can be conditional or unconditional.
40. **Multi-Function Unit**: Unit having the capability to be programmed for specific use and time-of-use constraints.
41. **Single-Function Unit**: is a Unit that is non-programmable and has no ability constrain use.
42. **Unit Use Restriction**: constrains what the Unit can be used for.
43. **Unit Origin Restriction**: constrains where, what the Unit can be used for, originates.
44. **Unit Time Restriction**: constrains when the Unit can be used.
45. **Unit Location Restriction**: constrains where the Unit can be used.
46. **Unit Denomination Adjustable**: defines how a Unit can be divided into smaller or fractional sub-Units.
47. **Unit Value Drivers**: the mechanism by which the Unit derives its Unit Value.
48. **Initial Value Offer**: refers to the Unit Value Offer by the Issuer in the Genesis Event.
49. **Extrinsic Unit Value Drivers:** mechanisms that are externally *guaranteed* through *Pegging* and backed by Collateral.
50. **Intrinsic**: mechanisms that are inherently *guaranteed by law or through contract.*
51. **Issuer Guaranteed Driver**: 1) Public Sector as Central Bank, or Government, or 2) Private Sector as Private Bank, or Corporate Entity, or 3) Backer.
52. **Backer Degree-of-Separation**: the number of parties as potential Claimants between the Backer and the Borrower referred to as Degree-of-Separation as Integers: 1, 2, 3…
53. **Non-Fungible Tokens**: representations of Value that cannot be copied, substituted, or subdivided, referred to as Collectables.
54. **Backing Collateral Requirement**: specifies whether the Backing will require Collateral (x 1.0) or not (x 0) and the percentage risk factor to apply (Backing Amount x 1.#).
55. **Backing Collateral Type**: specifies the type of Collateral required when Backing Collateral Requirement is 1.#. Types include Commodities, Securities, Digital Currency, Digital Assets
56. **External Pegging Mechanism**: the mechanism by which the Value of a Unit is determined by its direct association or pegging to an external Value source.
57. **On-Chain Lock-in**: represents how a Value Amount is locked and retained as collateral on a distributed ledger-technology (DLT) blockchain-based DC, referred to as a Cryptocurrency.
58. **Escrow**: represents how Value Amount is unavailable and held as collateral for non-blockchain- and off-blockchain-based DC use cases.

8.2 Market ontology

The following is a set of definitions for terms related to digital currency, market participants, and transactions. These include definitions for different types of participants in the market, such as issuers, owners, and intermediaries; terms related to the supply and value of digital currency units; and terms related to various types of transactions, such as lending, exchanging, and validating. The text also includes definitions for concepts such as proof of identity, authentication, and authorization, and introduces concepts such as "distributed autonomous organization" and "oracle" in the context of digital currency markets.

1	Market
1.1	Issue Amount Participant
1.1.1	Genesis Supply Event
1.1.2	Other Supply Events
1.2	Owner of Amount
1.2.1	Access to Units in Store is Ownership
1.2.2	Amount Value Change Over Time with no Activity
1.2.3	Credit Amount
1.3	Other Market Participants
1.3.1	Other Existing Amount Owner Participants
1.3.2	Other Non-Owner Participants
1.4	Market Engagement
1.4.1	Positive Engagement
1.4.2	Negative Engagement

Figure 12: Digital Currency Market Ontology: Level 1 and 2

1	Market
1.1	Issue Amount Participant
1.1.1	Genesis Supply Event
1.1.2	Other Supply Events
1.2	Owner of Amount
1.2.1	Access to Units in Store is Ownership
1.2.1.1	Proof-of-Identity
1.2.1.1.1	Identification
1.2.1.1.2	Authentication
1.2.1.1.3	Authorization
1.2.1.2	Proof-of-Control
1.2.1.2.1	Send Action
1.2.1.2.2	Trusted Intermediary
1.2.2	Amount Value Change Over Time with no Activity
1.2.2.1	% Change in Amount Value
1.2.3	Credit Amount
1.2.3.1	% of Amount Owned
1.3	Other Market Participants
1.3.1	Other Existing Amount Owner Participants
1.3.1.1	Other Owners Like Me
1.3.1.2	Financial Intermediary Participants
1.3.1.2.1	Liquidity Participants
1.3.2	Other Non-Owner Participants
1.3.2.1	Service Participant Non-Owners
1.3.2.1.1	Code Participants
1.3.2.1.2	Service Participants
1.3.2.1.3	Oracle Participants
1.3.2.2	Reward Participants: New Amount created
1.4	Market Engagement
1.4.1	Positive Engagement
1.4.1.1	Commit to Change Ownership
1.4.1.1.1	My Commit to Transfer
1.4.1.1.2	My Commit to Purchase Asset or Service
1.4.1.1.3	Other Commit to Sell Asset
1.4.1.1.4	Commit with No Change in Ownership
1.4.2	Negative Engagement
1.4.2.1	Commit to Accept Other Amount Ownership:
1.4.2.1.1	My Commit to Borrow and Amount Deposited
1.4.2.2	Uncommit My Amount Ownership or Control
1.4.2.2.1	Uncommit Other Owned Value under Control
1.4.2.2.2	Uncommit My Owned Value

Figure 13: Digital Currency Market Ontology: Level 3

1	Market
1.1	Issue Amount Participant
1.1.1	Genesis Supply Event
1.1.2	Other Supply Events
1.2	Owner of Amount
1.2.1	Access to Units in Store is Ownership
1.2.1.1	Proof-of-Identity
1.2.1.1.1	Identification
1.2.1.1.2	Authentication
1.2.1.1.3	Authorization
1.2.1.2	Proof-of-Control
1.2.1.2.1	Send Action
1.2.1.2.2	Trusted Intermediary
1.2.2	Amount Value Change Over Time with no Activity
1.2.2.1	% Change in Amount Value
1.2.3	Credit Amount
1.2.3.1	% of Amount Owned
1.3	Other Market Participants
1.3.1	Other Existing Amount Owner Participants
1.3.1.1	Other Owners Like Me
1.3.1.2	Financial Intermediary Participants
1.3.1.2.1	Liquidity Participants
1.3.1.2.1.1	Lender
1.3.1.2.1.2	Collateral Provider
1.3.2	Other Non-Owner Participants
1.3.2.1	Service Participant Non-Owners
1.3.2.1.1	Code Participants
1.3.2.1.1.1	Distributed Autonomous Organizations
1.3.2.1.1.2	Contracts (Move Terms)
1.3.2.1.2	Service Participants
1.3.2.1.2.1	Exchange Service
1.3.2.1.2.2	Transfer Services
1.3.2.1.2.3	Custodians (Amount Holders)
1.3.2.1.3	Oracle Participants
1.3.2.1.3.1	Prices
1.3.2.1.3.2	Events
1.3.2.2	Reward Participants: New Amount created
1.3.2.2.1	Validators
1.3.2.2.2	Verifiers
1.4	Market Engagement
1.4.1	Positive Engagement
1.4.1.1	Commit to Change Ownership
1.4.1.1.1	My Commit to Transfer
1.4.1.1.1.1	My Commit to Lend =|
1.4.1.1.1.2	My Commit to Discharge Debt =|
1.4.1.1.1.3	My Commit to Invest = |
1.4.1.1.1.4	My Commit to Sell = |
1.4.1.1.2	My Commit to Purchase Asset or Service
1.4.1.1.4.1	Transact for Digital Assets & Digital Serve
1.4.1.1.4.2	Transact for Physical Asset
1.4.1.1.3	Other Commit to Sell Asset
1.4.1.1.4	Commit with No Change in Ownership
1.4.1.1.1.1	Commit to Exchange [Change Unit Form @ constan
1.4.1.1.1.2	Commit to Stake
1.4.2	Negative Engagement
1.4.2.1	Commit to Accept Other Amount Ownership:
1.4.2.1.1	My Commit to Borrow and Amount Deposited =|
1.4.2.2	Uncommit My Amount Ownership or Control
1.4.2.2.1	Uncommit Other Owned Value under Control
1.4.2.2.2	Uncommit My Owned Value
1.4.2.2.2.1	Move Ownership to Cold storage (Disconnected Val
1.4.2.2.2.2	Un-Stake

Figure 14: Digital Currency Market Ontology: Level 5

1. **Participant:** An entity having control over a one or more Stores in one or more Ecosystems that can participate in an Agree.
2. **Issuer Participant**: an entity that has the capability to issue Form.
3. **Owner Participant**: A Participant that "owns" an Amount of Value
4. **Non-Owner Participant**: A Participant that "does not own" an Amount
5. **Entrant Participant:** Potential future Owner
6. **Service Participant**: Participant that provides Services-for-a-Fee, Fee originating from Existing Supply.
7. **Reward Participant**: Participant that provides Services-for-a-Reward, Renumeration originating from new additional Supply
8. **Fee:** Amount a Service Provider will extract (by paid) from the transaction for the service provided.
9. **Reward:** A Participant that generates validation trust in decentralized consensus mechanisms. Reward originating from newly issued Form.
10. **Genesis Supply**: is the 1st Supply event of Units produced by Issuer to establish the Unit Value of the Unit, referred to as the Initial Value Offer.
11. **Subsequent Supply Events** refers to all Issuer Supply events after Genesis Supply.
12. **Amount Owner**: Amount being the Unit Value times the number of Units owned by a Participant.
13. **Unit Access**: the control by a Participant that can be exercised on the Units owned either through an authentication or proof of control.
14. **Proof-of-Identity**: a process of establishing the authentic identity of a Participant to a level of assurance and whether the person asserting the identity is who they say they are.
15. **Identification**: the process of establishing the true identity of a Participant through a background vetting process.
16. **Authentication**: the process of establishing that the person making the Identity assertion is who they say they are.
17. **Authorization**: the process of controlling and constraining the use of the Unit based on rights management.
18. **Proof-of-Control**: the ability to demonstrate Unit Access and control not through Identification but through action.
19. **Send Action**: proof-of-control demonstrated by a transfer of the Unit to the control of a Receiver Participant.
20. **Trusted Intermediary**: A Participant that provides a Validation and/or Verification Service to other Participants.
21. **Amount Depreciation**: Amount of Value loss over time with no Activity
22. **Amount Value Percent Loss**: is a measure of depreciation.
23. **Credit Amount Available**: An Amount of Value available as Credit when exercised becomes a Debt.
24. **Percent of Amount Owned** is a measure of Credit Amount Available
25. **Financial Intermediary** is a Participant acting as "go-between" two or more Participants.
26. **Liquidity Participant** is one that Owns Amount and make it available for Borrowing.
27. **Lender** is a Liquidity Participant that performs Lending Transactions with Borrowers for an **Interest Rat**e over a **Lending Period**.
28. Collateral Provider is a Liquidity Participant that provides Amount as a guarantee against a Loan.
29. Non-Owners are Participants that do not Own any Amount.
30. **Smart Contract:** Code Instructions contained in a program.

31. **Distributed Autonomous Organization** is a Participant whose conduct and actions are based on a Consensus Algorithm with Oracle data inputs.
32. **Exchange** Service ("Exchange") is a Participant that converts one Unit Form for another based on their Unit Value ratio, referred to as **Exchange Rate**.
33. Transfer Service is an **Intermediary Participant** that Moves Amount from a **Source Store** owned by a **Sender** Participant to a **Destination Store** owned by a **Receiver** Participant.
34. **Custodian** is a Participant that "holds" an Amount on behalf of the Owner.
35. **Oracle** is a Participant that provides objective and empirical data as a service to Participants such as DAOs (Decentralized Autonomous Organizations) and Exchanges.
36. Price: is the Amount paid, or to be paid, in a **Purchase Transaction**.
37. **Validator**: is a Participant that provides validation/confirmation Service-for-Reward.
38. **Verifiers**: is a Participant that provides a verification, or proof certification Service-for-Reward.
39. **Market Engagement**: The commit of a **Participant** and another Market Participant, involving an Amount, in one or more Forms.
40. **Positive Engagement**: The commit made by a **Participant** involves adding liquidity to the Market.
41. **Change Ownership Commit**: The commit made by a **Participant** involves a change of Ownership of Amount committed.
42. **Transfer Commit**: The commit made by a **Participant** involves the transfer of Amount to a **Receiving Participant**.
43. **Exchange Commit**: Commit to Exchange one Form for another based on their constant Value Exchange Rate.
44. **Stake Commit**: Commit to Stake an Amount with no change in Ownership.
45. **Invest Commit**: Commit to Stake an Amount to Invest
46. **Sell Commit**: Offer to sell at a Price.
47. **Buy Commit**: Offer to purchase at Price.
48. **Digital Transact Commit:** Purchase Commit for Digital Asset and/or Digital Service
49. **Physical Transact Commit:** Purchase Commit for Physical Asset or Physical Service
50. **Negative Engagement**: The commit made by a **Participant** involves removing or locking liquidity rendering it unavailable as liquidity in the Market.
51. **Accept Commit**: The commit made by a **Participant** to Accept Ownership of Amount
52. **Cold Store**: Units maintained in Internet-disconnected Store.
53. **Stake**: To assign the right-of-Control of Amount to Liquidity Participant for Staking Period in return for renumeration.
54. **Un-Stake**: The act of un-assigning the previously assigned right-of-Control of Amount before the expiration of Staking Period.

8.3 Agree ontology

The following provides definitions for terms related to agreements ("Agree") made between participants in a digital currency ecosystem. These agreements can include lending, transferring, exchanging, staking, and buying assets and services. The text also defines different types of participants such as issuers, owners, and non-owners, and different types of agreements such as intra-ecosystem and inter-ecosystem agreements. The text also defines terms related to the terms and conditions of agreements, the initiation and fulfilment of agreements, and the movement of assets and services.

1	**Agree**
1.1	**Agree Participants**
1.2	**Agree Terms**
1.3	**Agree Types**
1.4	**Agree Ownership Impact**
1.5	**Agree Finality**
1.6	**Store Availability**
1.7	**Commit-to-Destination Store Path**
1.8	**Update**

1	**Agree**
1.1	**Agree Participants**
1.1.1	Start Participant
1.1.2	One or more Intermediary Participants (Fees)
1.1.3	End Participant: Final Remainder Amount Rs = Full Amount - Fees
1.2	**Agree Terms**
1.2.1	Terms as Data
1.2.2	Terms as Code
1.3	**Agree Types**
1.3.1	Intra-Ecosystem
1.3.2	Inter-Ecosystem (-to-Market)
1.4	**Agree Ownership Impact**
1.4.1	Amount Ownership Restrictions
1.4.2	Amount Ownership Change
1.5	**Agree Finality**
1.5.1	Settle Now: Synchronously
1.5.2	Settle Later: Asynchronously
1.6	**Store Availability**
1.6.1	Commit Source Store Availability
1.6.2	Destination Store Availability
1.7	**Commit-to-Destination Store Path**
1.7.1	Intra-Ecosystem Path (within One)
1.7.2	Inter-Ecosystem Path (between Two)
1.7.3	Distributed Multi Ecosystem Path (between More then Two)
1.8	**Update**
1.8.1	No Ledger Update
1.8.2	One Unique Ledger Update
1.8.3	Many Distinct Ledger Updates
1.8.4	Physical Ledger Update
1.9	**Agree Definitions**
1.9.1	Agree Outcome Predictability
1.9.2	Agree Liquidity Impact

Figure 15: Digital Currency Agree Ontology: Level 1 and 2

1	Agree
1.1	Agree Participants
1.1.1	Start Participant
1.1.1.1	Agree to Transact
1.1.2	One or more Intermediary Participants (Fees)
1.1.2.1	"S" # Intermediary Participants, For I = 1 to S Intermediaries
1.1.3	End Participant: Final Remainder Amount Rs = Full Amount - Fees
1.1.3.1	Seller:
1.1.3.2	Borrower
1.1.3.3	Transferee
1.1.3.4	Exchanger
1.1.3.5	Liquidity Pool
1.2	Agree Terms
1.2.1	Terms as Data
1.2.2	Terms as Code
1.2.2.1	Code Input Data Sources
1.3	Agree Types
1.3.1	Intra-Ecosystem
1.3.1.1	Transfer (Move Amount)
1.3.2	Inter-Ecosystem (-to-Market)
1.3.2.1	Digital Asset Transaction (Ownership Change)
1.3.2.2	Physical Asset Transaction (Ownership Change)
1.3.2.3	Exchange [no Ownership Change]
1.4	Agree Ownership Impact
1.4.1	Amount Ownership Restrictions
1.4.1.1	Availability Restriction (Committed Value Amount not Available)
1.4.1.2	Staking (Cannot use for Period)
1.4.2	Amount Ownership Change
1.4.2.1	Full Amount
1.4.2.2	Commit Amount Sufficiency
1.5	Agree Finality
1.5.1	Settle Now: Synchronously
1.5.2	Settle Later: Asynchronously
1.5.2.1	By Design
1.5.2.2	By Context
1.6	Store Availability
1.6.1	Commit Source Store Availability
1.6.1.1	Accessible - Hot (Online)
1.6.1.2	Inaccessible - Cold (Offline)
1.6.2	Destination Store Availability
1.6.2.1	Accessible - Hot (Online)
1.6.2.2	Inaccessible - Cold (Offline)
1.7	Commit-to-Destination Store Path
1.7.1	Intra-Ecosystem Path (within One)
1.7.1.1	Direct Path
1.7.1.2	Indirect Path
1.7.2	Inter-Ecosystem Path (between Two)
1.7.3	Distributed Multi Ecosystem Path (between More then Two)
1.8	Update
1.8.1	No Ledger Update
1.8.2	One Unique Ledger Update
1.8.2.1	One Unique Centralized Ledger Update
1.8.2.2	Many Identical Distributed Ledger Updates
1.8.3	Many Distinct Ledger Updates
1.8.3.1	Many Unique Distinct Ledgers
1.8.4	Physical Ledger Update
1.9	Agree Definitions
1.9.1	Agree Outcome Predictability
1.9.1.1	Single Result Known Before
1.9.1.2	Variable Outcome, Result Value options Known, Result Unknown
1.9.1.3	Do Result
1.9.2	Agree Liquidity Impact:
1.9.2.1	Increasing Liquidity
1.9.2.2	Decreasing Liquidity

Figure 16: Digital Currency Agree Ontology: Level 3

1	**Agree**
1.1	**Agree Participants**
1.1.1	Start Participant
1.1.1.1	Agree to Transact
1.1.1.1.1	Lender: Commit Amount to Lend
1.1.1.1.2	Transferer: Commit Amount to Move Digital Store
1.1.1.1.3	Exchanger: Commit Amount to Exchange
1.1.1.1.4	Staking: Commit Amount to Stake
1.1.1.1.5	Buyer: Commit Amount to Purchase
1.1.2	One or more Intermediary Participants (Fees)
1.1.2.1	"S" # Intermediary Participants, For I = 1 to S Intermediaries
1.1.2.1.1	Full Amount = End Amount plus all Fees
1.1.2.1.2	Repeat For I = 2 to S, $R_i = R_{i-1} - F_i$
1.1.3	End Participant: Final Remainder Amount Rs = Full Amount - Fees
1.1.3.1	Seller:
1.1.3.1.1	Commit to send Physical Asset
1.1.3.1.2	Commit to send Digital Asset
1.1.3.2	Borrower
1.1.3.3	Transferee
1.1.3.4	Exchanger
1.1.3.5	Liquidity Pool
1.2	**Agree Terms**
1.2.1	Terms as Data
1.2.2	Terms as Code
1.2.2.1	Code Input Data Sources
1.3	**Agree Types**
1.3.1	Intra-Ecosystem
1.3.1.1	Transfer (Move Amount)
1.3.1.1.1	Sender (-) Value Amount from Source DCT Store | present value forwarc
1.3.1.1.2	Add (+) DCT Amount to Destination Store
1.3.2	Inter-Ecosystem (-to-Market)
1.3.2.1	Digital Asset Transaction (Ownership Change)
1.3.2.1.1	Remove/Decrease Unit Type 1 Amount from Buyer Unit Type Store | Co
1.3.2.1.2	Condiction: Unit Type 1 Amount/DA-V (set by Owner) = 1
1.3.2.1.3	Add/Increase Unit Type 1 Amount to Seller Store
1.3.2.1.4	Send Digital Asset to Buyer
1.3.2.2	Physical Asset Transaction (Ownership Change)
1.3.2.2.1	Remove/Decrease Unit Type 1 Amount from Buyer Unit Type Store | Co
1.3.2.2.2	Condition: Unit Type 1 Amount/PA-V (set by Owner) = 1
1.3.2.2.3	Add/Increase Unit Type 1 Amount to Seller Store
1.3.2.2.4	Send Physical Asset to Buyer
1.3.2.3	Exchange [no Ownership Change]
1.3.2.3.1	Remove/Decrease Unit Type 1 Amount from Exchanger Unit Type 1 Stor
1.3.2.3.2	Conversion rate: Unit Type 1 Value/Unit Type 2 Value
1.3.2.3.3	Add/Increase Unit Type 2 Amount to Exchanger Store
1.4	**Agree Ownership Impact**
1.4.1	Amount Ownership Restrictions
1.4.1.1	Availability Restriction (Committed Value Amount not Available)
1.4.1.2	Staking (Cannot use for Period)
1.4.1.2.1	Reward Amount Ownership
1.4.2	Amount Ownership Change
1.4.2.1	Full Amount
1.4.1.1.1	Destination Amount Required
1.4.1.1.2	Intermediary Fees Required
1.4.1.1.3	Applicable Taxes
1.4.2.2	Commit Amount Sufficiency
1.4.2.2.1	Full: Commit Amount = Required (Full) Amount, , when Commit Amoun
1.4.2.2.2	Under: Short Amount = Requested Amount - Commit Amount, , when C
1.4.2.2.3	Over: Unspent Transaction Output (UTXO), when Commit Amount > Full

Figure 17a: Digital Currency Agree Ontology: Level 4

1.5	Agree Finality
1.5.1	Settle Now: Synchronously
1.5.2	Settle Later: Asynchronously
1.5.2.1	By Design
1.5.2.2	By Context
1.6	Store Availability
1.6.1	Commit Source Store Availability
1.6.1.1	Accessible - Hot (Online)
1.6.1.1.1	
1.6.1.2	Inaccessible - Cold (Offline)
1.6.1.2.1	
1.6.1.2.2	
1.6.2	Destination Store Availability
1.6.2.1	Accessible - Hot (Online)
1.6.2.1.1	Internal Storage
1.6.2.2	Inaccessible - Cold (Offline)
1.6.2.2.1	Paper
1.6.2.2.2	External Storage
1.7	Commit-to-Destination Store Path
1.7.1	Intra-Ecosystem Path (within One)
1.7.1.1	Direct Path
1.7.1.1.1	collocated, same Store Space, Owner
1.7.1.2	Indirect Path
1.7.1.2.1	Custodian Managed Store
1.7.2	Inter-Ecosystem Path (between Two)
1.7.3	Distributed Multi Ecosystem Path (between More then Two)
1.8	Update
1.8.1	No Ledger Update
1.8.2	One Unique Ledger Update
1.8.2.1	One Unique Centralized Ledger Update
1.8.2.1.1	One Native Ledger
1.8.2.2	Many Identical Distributed Ledger Updates
1.8.2.2.1	Public Validators
1.8.2.2.2	Permissioned Validators
1.8.3	Many Distinct Ledger Updates
1.8.3.1	Many Unique Distinct Ledgers
1.8.3.1.1	Commit Amount Source Ledger
1.8.3.1.2	Intermediary Ledgers
1.8.3.1.3	Destination Amount Destination Ledger
1.8.4	Physical Ledger Update
1.9	Agree Definitions
1.9.1	Agree Outcome Predictability
1.9.1.1	Single Result Known Before
1.9.1.2	Variable Outcome, Result Value options Known, Result Unknown
1.9.1.3	Do Result
1.9.2	Agree Liquidity Impact:
1.9.2.1	Increasing Liquidity
1.9.2.2	Decreasing Liquidity

Figure 17b: Digital Currency Agree Ontology: Level 4

1	**Agree**
1.1	**Agree Participants**
1.1.1	**Start Participant**
1.1.1.1	**Agree to Transact**
1.1.1.1.1	**Lender: Commit Amount to Lend**
1.1.1.1.2	**Transferer: Commit Amount to Move Digital Store**
1.1.1.1.3	**Exchanger: Commit Amount to Exchange**
1.1.1.1.4	**Staking: Commit Amount to Stake**
1.1.1.1.5	**Buyer: Commit Amount to Purchase**
1.1.1.1.5.1	**A Physical Asset**
1.1.1.1.5.2	**A Digital Asset**
1.1.2	**One or more Intermediary Participants (Fees)**
1.1.2.1	**"S" # Intermediary Participants, For I = 1 to S Intermediaries**
1.1.2.1.1	**Full Amount = End Amount plus all Fees**
1.1.2.1.1.1	**Provide S1 Service, Extract Fee "F_1"**
1.1.2.1.1.2	**S1 Forward (Transfer) Remainder 1 to S2: R_1 = Full Amount - F_1**
1.1.2.1.2	**Repeat For I = 2 to S, $R_i = R_{i-1} - F_i$**
1.1.3	**End Participant: Final Remainder Amount Rs = Full Amount - Fees**
1.1.3.1	**Seller:**
1.1.3.1.1	**Commit to send Physical Asset**
1.1.3.1.2	**Commit to send Digital Asset**
1.1.3.2	**Borrower**
1.1.3.3	**Transferee**
1.1.3.4	**Exchanger**
1.1.3.5	**Liquidity Pool**
1.2	**Agree Terms**
1.2.1	**Terms as Data**
1.2.2	**Terms as Code**
1.2.2.1	**Code Input Data Sources**
1.3	**Agree Type composed of one or more Commit Swap Types**
1.3.1	**Intra-Ecosystem**
1.3.1.1	**Transfer (Move Amount)**
1.3.1.1.1	**Sender (-) Value Amount from Source DCT Store | present value forwar**
1.3.1.1.2	**Add (+) DCT Amount to Destination Store**
1.3.2	**Inter-Ecosystem (-to-Market)**
1.3.2.1	**Digital Asset Transaction (Ownership Change)**
1.3.2.1.1	**Remove/Decrease Unit Type 1 Amount from Buyer Unit Type Store | Co**
1.3.2.1.2	**Condiction: Unit Type 1 Amount/DA-V (set by Owner) = 1**
1.3.2.1.3	**Add/Increase Unit Type 1 Amount to Seller Store**
1.3.2.1.4	**Send Digital Asset to Buyer**
1.3.2.2	**Physical Asset Transaction (Ownership Change)**
1.3.2.2.1	**Remove/Decrease Unit Type 1 Amount from Buyer Unit Type Store | Co**
1.3.2.2.2	**Condition: Unit Type 1 Amount/PA-V (set by Owner) = 1**
1.3.2.2.3	**Add/Increase Unit Type 1 Amount to Seller Store**
1.3.2.2.4	**Send Physical Asset to Buyer**
1.3.2.3	**Exchange [no Ownership Change]**
1.3.2.3.1	**Remove/Decrease Unit Type 1 Amount from Exchanger Unit Type 1 Stor**
1.3.2.3.2	**Conversion rate: Unit Type 1 Value/Unit Type 2 Value**
1.3.2.3.3	**Add/Increase Unit Type 2 Amount to Exchanger Store**
1.4	**Agree Activity Impact on Ownership**
1.4.1	**Amount Ownership Restrictions**
1.4.1.1	**Availability Restriction (Committed Value Amount not Available)**
1.4.1.2	**Staking (Cannot use for Period)**
1.4.1.2.1	**Reward Amount Ownership**
1.4.2	**Amount Ownership Change**
1.4.2.1	**Full Amount**
1.4.1.1.1	**Destination Amount Required**
1.4.1.1.2	**Intermediary Fees Required**
1.4.1.1.3	**Applicable Taxes**
1.4.2.2	**Commit Amount Sufficiency**
1.4.2.2.1	**Full: Commit Amount = Required (Full) Amount, , when Commit Amoun**
1.4.2.2.2	**Under: Short Amount = Requested Amount - Commit Amount, , when C**
1.4.2.2.2.1	**Debt Created (Short Amount)**
1.4.2.2.2.2	**Short Debt Terms**
1.4.2.2.3	**Over: Unspent Transaction Output (UTXO), when Commit Amount > Full**
1.4.2.2.3.1	**UTXO = Requested (Full) Amount - Commit Amount**

Figure 18a: Digital Currency Agree Ontology: Level 5 and higher

1.4.2.1	Full Amount
1.4.1.1.1	Destination Amount Required
1.4.1.1.2	Intermediary Fees Required
1.4.1.1.3	Applicable Taxes
1.4.2.2	Commit Amount Sufficiency
1.4.2.2.1	Full: Commit Amount = Required (Full) Amount, , when Commit Amoun
1.4.2.2.2	Under: Short Amount = Requested Amount - Commit Amount, , when C
1.4.2.2.2.1	Debt Created (Short Amount)
1.4.2.2.2.2	Short Debt Terms
1.4.2.2.3	Over: Unspent Transaction Output (UTXO), when Commit Amount > Full
1.4.2.2.3.1	UTXO = Requested (Full) Amount - Commit Amount
1.5	Agree Finality
1.5.1	Settle Now: Synchronously
1.5.2	Settle Later: Asynchronously
1.5.2.1	By Design
1.5.2.2	By Context
1.6	Store Availability
1.6.1	Commit Source Store Availability
1.6.1.1	Accessible - Hot (Online)
1.6.1.1.1	
1.6.1.2	Inaccessible - Cold (Offline)
1.6.1.2.1	
1.6.1.2.2	
1.6.2	Destination Store Availability
1.6.2.1	Accessible - Hot (Online)
1.6.2.1.1	Internal Storage
1.6.2.2	Inaccessible - Cold (Offline)
1.6.2.2.1	Paper
1.6.2.2.2	External Storage
1.7	Commit-to-Destination Store Path
1.7.1	Intra-Ecosystem Path (within One)
1.7.1.1	Direct Path
1.7.1.1.1	collocated, same Store Space, Owner
1.7.1.2	Indirect Path
1.7.1.2.1	Custodian Managed Store
1.7.2	Inter-Ecosystem Path (between Two)
1.7.3	Distributed Multi Ecosystem Path (between More then Two)
1.8	Update [Amount Ownership Changes]
1.8.1	No Ledger Update
1.8.2	One Unique Ledger Update
1.8.2.1	One Unique Centralized Ledger Update
1.8.2.1.1	One Native Ledger
1.8.2.2	Many Identical Distributed Ledger Updates
1.8.2.2.1	Public Validators
1.8.2.2.2	Permissioned Validators
1.8.3	Many Distinct Ledger Updates
1.8.3.1	Many Unique Distinct Ledgers
1.8.3.1.1	Commit Amount Source Ledger
1.8.3.1.2	Intermediary Ledgers
1.8.3.1.3	Destination Amount Destination Ledger
1.8.4	Physical Ledger Update
1.9	Agree Definitions
1.9.1	Agree Outcome Predictability
1.9.1.1	Single Result Known Before
1.9.1.2	Variable Outcome, Result Value options Known, Result Unknown
1.9.1.3	Do Result
1.9.2	Agree Liquidity Impact:
1.9.2.1	Increasing Liquidity
1.9.2.2	Decreasing Liquidity

Figure 18b: Digital Currency Agree Ontology: Level 5 and higher

The following are the Agree related definitions.

133. **Commit**: the act by a Participant in an Offer to buy or sell.
134. **Start Participant**: Participant that initiates a Commit.
135. **Amount**: the Unit Value times the number of Units.
136. **Agree**: covers all activities and actions involved in an Agreement ("Agree") initiated by a Start Participant with an Amount Commit.
137. **Agree Participants**: all Participants involved in one Agree.
138. **Lender**: A Start Participant that Commits Amount to Lend
139. **Transferer**: A Start Participant that Commits Amount to Move from one Digital Store to another.
140. **Exchanger**: A Start Participant that Commits Amount to Exchange
141. **Staker**: A Start Participant that Commits Amount to Stake
142. **Buyer**: A Start Participant that Commits Amount to Purchase
143. **Physical Asset/Service**: an object or Service of value that exists or is provided in the natural world.
144. **Digital Asset/Service**: an electronically based Service and a digital cryptographic object both of value that exists or is provided in the virtual world.
145. **Intermediary Participant Pay-it-Forward (Serial) Fee Model**
"S" = # Intermediary Participants, For I = 1 to S, Commit Full Amount = End Amount plus all Fees. S1 Service Provided, Fee "F1" extracted.

S1 Forwards (Transfers) Remainder 1 to S2: where R1 = Full Amount - F1. Repeat For I = 2 to S, Ri = Ri-1 – Fi.

146. **End Participant** receives Final Remainder Amount Rs = Full Amount – all Fees
147. **Seller**: Participant that offers Physical or Digital Assets/Services
148. **Physical Asset Send**: An Agree involving the Purchase of a Physical Asset that must be sent to **Buyer**.
149. **Digital Asset Send**: An Agree involving the Purchase of a Digital Asset Object that must be sent to **Buyer**.
150. **Borrower**: A Participant that accepts a Loan from a Lender under terms set in Loan Agree.
151. **Liquidity Pool**: a reserve of Amount available for lending.
152. **Agree Terms**: the contractual terms and conditions of an Agree.
153. **Terms of Data** ("Data"): the contractual terms and conditions of an Agree encoded as data in a File.
154. **Terms of Code** ("Code"): the contractual terms and conditions of an Agree encoded as computer instructions.
155. **Agree Classes**: all Agrees are either Intra-Ecosystem, conducted within same Form of Value, or Inter-Ecosystem, between two or more Ecosystems of different Forms of Value.
156. **Intra-Ecosystem Agree**: Agree executed within same Form of Value.
157. **Transfer**: Start Participant moves an Amount of Form from **Sender Store** to **Receiver Store**. Sender Commits (Lock) Amount from source **Sender Store**, which is unavailable to Sender during Lock Period. Amount less Fees added to destination **Receiver Store**
158. **Inter-Ecosystem Agree**: Agree executed between two or more different Forms of Value through no, or one or more **Intermediary Participants**
159. **Digital Asset/Service Transaction** is an Agree to Purchase a Digital Asset/Service in exchange for a **Purchase Amount** equal to **Seller Price + Fees**.
~~160.~~ **Agree Initiation**: Start Participant as Buyer Commits (Lock) Purchase Amount from **Buyer Store**, unavailable to Buyer during **Lock Period**.
161. **Agree Condition**: (Purchase Amount-Fees)/Seller Price = 1

162. **Agree Fulfilment**: If yes, add/increase Seller Price to Seller Store. Seller to send Digital Asset to, or provide Digital Service to Buyer
If no, Agree is terminated and Purchase Amount unlocked and returned to Buyer Store.
163. **Physical Asset/Service Transaction**: is an Agree to Purchase a Physical Asset/Service in exchange for a **Purchase Amount** equal to **Seller Price + Fees**.
164. **Agree Initiation**: Start Participant as Buyer Commits (Lock) Purchase Amount from **Buyer Store**, unavailable to Buyer during **Lock Period**.
165. **Agree Condition**: (Purchase Amount-Fees)/Seller Price = 1
166. **Agree Fulfilment**: If yes, add/increase Seller Price to Seller Store. Seller to send Physical Asset to, or provide Physical Service to Buyer
If no, Agree is terminated and Purchase Amount unlocked and returned to Buyer Store.
167. **Exchange**: is an Agree to convert a **Source Unit Form** for an **Exchange Unit Form** based on the Unit Value ratio of the two Forms.
168. **Agree Initiation**: Start Participant as Exchanger Commits (Lock) **Exchange Amount** of Source Unit Form from **Exchanger Source Store**, unavailable to Exchanger during **Lock Period**.
169. **Agree Condition**: Exchange Rate = Unit Value of Source Form/Unit Value of Exchange Form, Exchange Fee
170. **Agree Fulfilment**: If yes, add/increase in **Exchanger Destination Store** by **Destination Amount** = Exchange Amount – Exchange Fee
171. **Agree Ownership Impact**: the outcome on ownership of Amount committed in Agree ("**Agree Amount**") as either full ownership change or no ownership change with the addition of restrictions for Restriction period.
172. **Ownership Restrictions:** are constraints on the Owner on the exercising the Amount Ownership rights, sale for a **Restriction Period**.
173. **Amount Availability Restriction**: the constraint on Amount is its availability.
174. **Agree Tax**: is the Amount of the Commit Amount extracted as a Tax.
175. **Amount Sufficiency**: whether the **Commit Amount** meets the **Agree Amount**
176. **Short Amount**: when Amount Sufficiency is negative, Commit Amount is less and does not meet the Agree Amount, a Debt Amount is generated where Start Participant accepts a Loan for the Short Amount
177. **Agree Finality**: refers to the temporal nature of Agree, whether it is reconciled synchronously in real time, or reconciled Asynchronously.
178. **Settle Now Synchronously**: refers to reconciling Agree synchronously in real time.
179. **Store Availability**: in both the Issuance Process and the Move Processes involve Stores that must be available for the Agree to process.
180. **Settle Later Asynchronously**: refers to reconciling Agree asynchronously by choice or design; by unintentional events; and by the unavailability of Stores in Agree.
181. **By Design Asynchronous Settlement**: Agree initiated is to be reconciled at more than one time as part of the **Agree Terms**
182. **By Context Asynchronous Settlement**: Agree initiated is reconciled later due to an unexpected Agree event.
183. **By State Asynchronous Settlement**: Agree initiated is reconciled later due to Store in Agree not being available.
184. **Accessible Online Store**: Store that is connected to the Internet
185. **Inaccessible Offline Store**: Store that is not connected to the Internet
186. **Paper**: A Form of a Unit in the Physical Asset.
187. **Commit-to-Destination Store Path**: are the connections necessary to complete the Amount movements between Agree participants as prescribed in Agree.
188. **Intra-Ecosystem Store Path**: all connections necessary to complete the Amount movements between Agree participants are in the same Ecosystem.
189. **Direct Intra-Ecosystem Store Path:** connections necessary to complete the Amount movements between Agree participants are in the same **Physical Store Space**.

190. **Indirect Intra-Ecosystem Store Path:** connections necessary to complete the Amount movements between Agree participants are in the separate **Physical Store Spaces**.
191. **Custodian** Store: A Store being managed by a Service Participant on behalf of an Owner.
192. **Inter-Ecosystem Path**: connections necessary to complete the Amount movements between Agree participants are in two different Ecosystems.
193. **Distributed Multi-Ecosystem Path** connections necessary to complete the Amount movements between Agree Participants are in the more than two Ecosystems.
194. **Ownership Change Update**: refers to recording all Amount Ownership changes involved in Agree.
195. **No Ledger Update**: refers to an Agree involving Forms where no update of Owner in Amount is recorded.
196. **One Unique Ledger Update**: An Agree involving Forms where the update of Amount Ownership change is recorded on one unique **Ledger**.
197. **One Unique Centralized Ledger Update**: An Agree involving Forms where the update of Amount Ownership change is recorded on one centralized **Ledger**.
198. **One Native Ledger**: An Agree involving Forms where the update of Amount Ownership change is recorded on one **Native Ledger**, one where the Form of Unit is native or of same protocol as the distributed ledger technology (DLT) recording mechanism.
199. **Many Identical Distributed Ledger Updates**: An Agree involving Forms where the update of Amount Ownership change is recorded on many identical distributed Ledgers.
200. **Public Validators**: Reward Participants providing a consensus service-for-renumeration in a public permissionless DLT-based network.
201. **Permissioned Validators**: Reward Participants providing a consensus service-for-renumeration in a private permissioned DLT-based network.
202. **Many Distinct Ledger Updates**: An Agree involving Forms where the update of Amount Ownership change is recorded on many distinct Ledgers.
203. **Many Unique Distinct Ledgers**: An Agree involving Forms where the update of Amount Ownership change is recorded on many unique Ledgers.
204. **Commit Ledger Update**: the ledger type of the Start Participant initiating the Agree.
205. **Intermediary Ledger Update**: the ledger type of the Intermediary Participants involved in the Agree.
206. **Destination Ledger Update**: the ledger type of the End Participant completing the Agree.
207. **Physical Ledger Update**: An Agree involving Forms where the update of Amount Ownership change is recorded on Paper.