Constitution of the Open Currencies System

October 13, 2019

A General Principles

The voluntary community of persons formed in accordance with the rules and procedures of the Open Currencies System (OCS) consists of the following three types of participants:

- The Operators: These members conduct economic operations (e.g. production and/or sales) which trigger the process of liquidity creation and destruction as described below.
- The Referees: The referees regulate the process of liquidity creation by either approving or rejecting proposals for operations put forth by individual operators.
- The Notaries: The notaries validate all transactions, thereby confirming that a proposed transaction is formally consistent with the constitution of the OCS. Furthermore, notaries disseminate new information.

The system contains a varying set of available currencies, which we refer to as OCS currencies for short, which are used by the operators to conduct their economic operations (e.g. trade). For any OCS currency and any given moment there is a limited volume of liquidity, i.e. units of amount of the respective currency, in circulation. For any OCS currency liquidity is created and destroyed through the following mechanism:

1. An operator writes a *proposal for an operation* he or she intends to conduct. As part of this proposal he or she requests a certain amount of liquidity of the chosen currency.
2. If the request is granted by the referees the operator is obliged from this moment on to conduct the operation as described in the proposal and uses the liquidity provided to this end. This liquidity is created at the moment the request is granted and brought into circulation by the operator as he or she makes purchases needed for the operation.
3. Every operation ends with the provision of goods and services in some quantity as described in the proposal and the receipt of an amount of liquidity by the operator depending on the highest total price to which these goods and services can be currently sold. In other words, the operator sells the final product to whoever pays the most and obtains some liquidity of the respective OCS currency in return.
4. Finally, the liquidity obtained at the end of the operation is permanently deleted from the system by the operator, thereby negating the creation of liquidity at the beginning of the operation. The time of this final step is referred to as maturity.

Since all liquidity is created through this process the current total volume of liquidity in circulation for a given currency corresponds to the total volume of ongoing economic operations (e.g. production), which will ultimately lead to sales of goods and services in this currency in the future. When new operations are initiated the total volume of liquidity increases, but shrinks when these

operations are concluded. What gives actual value to a currency is the fact that the operators are obliged to seek to obtain as much of the currency as possible in return for the goods and services they provide at the end.

Before going through the guidelines and procedures to be followed by individual members of the community (section B) and details regarding the technical realization of the system (section C), the fundamental principles and concepts, these rules and procedures are designed to implement, are hereby listed:

Transfer of Risk: Operators assume economic rather than financial obligations. An operator who receives a certain amount of liquidity to conduct the operation he or she outlined in the proposal is not responsible for paying back the same amount of liquidity at the end of the operation. Instead, the operator is responsible for fulfilling his or her obligations as defined in the operation proposal and must seek to obtain as much liquidity as possible at the conclusion of the operation. In particular, the operator is not subject to the price fluctuation risk: If the price of the goods provided by the operator decreases and the goods or services are sold at a lower price than initially expected, a lower amount of liquidity is deleted, without the operator having to make up for the gap between the created and the deleted liquidity. Thus, this risk is effectively transferred to the community by means of a possible liquidity surplus. At the same time the operator is responsible for providing complete and objective information in the proposal based on what is known to him or her to allow the referees to make the right choice and is not allowed to withhold information.

Openness: The OCS is open to everyone who is capable and willing to follow the constitution of the OCS and be a productive member of the community. A prospective referee or notary cannot be rejected on any ground other than his or her ability and/or intent to fulfill his or her prospective functions as outlined in this document. Similarly an operation proposal must be approved if and only if the requested amount is justified by the proposal in light of the currency's description. A proposal can only be rejected on the grounds of not fulfilling the requirements defined in this document or a legitimate lack of trust in the operator's ability and/or intent to conduct the operation as planned and cannot be rejected for any other reason.

Decentralization: The information regarding decisions made by members and regarding transactions that have taken place is not stored centrally in any specific central location. Essentially, the validity of a transaction or decision can be verified based on the existence of a sequence of informational building blocks which are subsequent and well-connected according to definitions of section C and trace back to the initial building blocks which are fixed and known to everybody. This sequence of buildings blocks serves as proof that a transaction has taken place and is valid and can be stored locally by anyone who is interested in being able to prove this in the future.

The OCS is also decentralized on the organizational level: There is no central governing body or regulator. Regulation occurs implicitly by the referees and/or the notaries, who are a loose community of persons making decisions independently of each other and overseeing each other's activities. Referees and notaries are expected to be impartial when making decisions and avoid coordinating their activities outside the procedures mentioned in this constitution.

Co-optation: New referees and notaries are appointed by acting referees and notaries according to the procedures described in section B. The initial referees and the initial notaries of the system are identified by public keys which are fixed and known to everybody. All other referees and notaries can be identified as such by a sequence of appointments tracing back to the initial referees or notaries. All referees and notaries have limited tenure during which they fulfill various functions assigned to them in accordance with the constitution. After that a referee or a notary must be reappointed by acting referees or notaries respectively to resume his or her activities.

The constitution of the OCS is an implied but binding contract entered by everybody who decides to take part in the OCS. In other words, by seeking to become a notary, a referee or an operator within the OCS the respective person agrees to act in accordance with this constitution and encourage other members of the community to do so as well, while abstaining from activities which are in contradiction to it or which are such that other participants are incited to act contrary to this constitution. The same applies to individuals who instruct or seek to instruct prospective notaries, referees or operators, as well as individuals who develop or seek to develop software to be used by prospective notaries, referees or operators.

If any provisions of this document become or turn out to be unimplementable or impractical then the respective provisions are to be replaced by such provisions which are implementable and practical, fulfill the functions which the unimplementable or impractical provisions were expected to fulfill and, under these constraints, are as close to the unimplementable or impractical provisions as possible. The same applies to any provisions which turn out to be inaccurate, incomplete, inconsistent with the intent of the constitution or insufficient to fulfill the respective purpose as stated in the document or implied by the context.

It is prohibited to alter this document and distribute the altered version without pointing out that it is not the original. It is likewise prohibited to provide or to distribute misleading information about the content of the constitution of the OCS.

B Rules, Guidelines and Procedures

B.1 Description of a currency

The description of a new currency must contain the following information:

- A definition of the minimal and maximal expected duration of operations under this currency.
- A definition of the basic rate at which all outstanding liquidity is deleted (as annual percentage).
- A definition of the minimal transfer amount in this currency for standard transfers.
- A definition of the minimal and maximal fee (measured in the currency itself) for an operation approval (see section B.3) depending on the requested amount.
- A definition of the referee stake (measured in the currency itself) in an operation proposal review process (see section B.3) depending on the requested amount and/or the fee.
- A definition of the minimal and maximal processing time for operation proposals (see section B.3).
- The public key of the first referee.
- A definition of the stake in a referee application review process (see section B.4) for this currency.
- A definition of the processing time for one step in a referee application review process (see section B.4).
- A definition of the maximal number of new referees an acting referee can appoint.
- A definition of the time a new referee serves before having to be reappointed.
- A definition of the amount of liquidity provided to every newly appointed referee.

- The maximal total amount of liquidity that can be created by one referee via approval of operation proposals during his or her tenure.
- A definition of the percentage of all profits of a referee which must go to the referee who appointed him or her.
- A set of *guidelines for the evaluation of operation proposals* sufficient to determine how much liquidity can be created based on expectations regarding the net economic output of an operation.
- The description may also specify standards for appointment of new referees (in addition to what the constitution states) to enforce the prescriptions of the constitution for this currency in practice.

The aforementioned guidelines for the evaluation of operation proposals contain explicitly or implicitly the definition of the *economic purpose* of the currency as well as the definition of the *units of account* in which the value, from the point of view of this economic purpose, of different goods and services is measured:

An economic purpose can for instance be to provide nutrients to as many natural persons as possible to keep them well-fed for as long as possible. Accordingly, different economic operations can be compared based on the nutritional value implicitly provided or associated with the goods or services provided at the end of a given operation. Note that any good or service can be seen to have some utility from the point of view of the aforementioned purpose, since even an unrelated good can be usually exchanged into something edible. However, since such a hypothetical transaction is subject to uncertainty and additional costs, the expected or estimated utility must be somewhat discounted if an economic operation is not directly related to a given purpose, especially for operations with high maturity. We refer to this as *uncertainty discount*.

A unit of account may, for instance, be *calories*, or *barrels of oil, ounces of silver, man-hours* etc., but also *Euros, Dollars* etc. In the last two examples, however, the purpose and the economic value behind a unit cannot be straightforwardly implied from the context and may not be well-defined, s.t. additional definitions and clarifications would be necessary.

A practical way to define the value of different possible goods and services, measured in the chosen units of account, is to use so-called *normal prices*. Under this approach one would simply list various typical goods and services which might appear in operation proposals as the expected economic output and then attach to each item its normal price. This normal price is not necessarily the actual market price but a hypothetical fair price as a reflection of the item's economic value. Here a "good or service" can also be a well-defined portfolio of goods and services. If now an operation proposal contains in its envisaged output only items for which normal prices are already defined and known, then the overall value of the output of a given operation is, assuming there is no uncertainty regarding the output and no side effects, straightforward to calculate as the *sum of the normal prices of these items*.

An operation proposal may, however, imply the provision of goods and services which are not explicitly mentioned in the description of the currency, such that normal prices for these goods or services are not directly defined. They can be estimated indirectly by comparing unmentioned goods or services to those which are mentioned in the description:

Ratio comparison: Assume a good or service A is explicitly mentioned in the description and the normal price is defined. If a good or service B is not mentioned, but can be used for the same purpose as A, one can estimate its normal price by scaling the price of A with the number of goods or services A used for a purpose divided by the minimal number of goods and

services B which can also be used to the same end with the same effect. Let x be this ratio by which the price of A is multiplied. x might not be known precisely and only an estimate might be available. Let's say the ratio is $x \pm \varepsilon$, where ε is an estimated standard deviation. Then, the price of B discounted for ratio uncertainty is the price of A multiplied with $x - \alpha \cdot \varepsilon$, where α is the *ratio uncertainty discount factor*, which is characteristic for a given currency and may be mentioned in its description under the uncertainty discount standards.

Exchange comparison: Assume it is not feasible to use a good or a service B directly for the purpose mentioned in the currency description. In order to estimate its normal price under the currency one may assume that the good or service may be exchanged in the market in the future for a good or service A for which a normal price was already estimated or defined. However, the ratio of the actual future prices of B and A is subject to uncertainty. Let us assume the ratio is $x \pm \varepsilon$, where x is the average expected price ratio and ε is an estimated standard deviation. In most cases one may use the current price ratio for the two goods on the market for x and estimate ε based on how strongly prices fluctuate and on what the time to maturity is expected to be. One may now estimate the normal price for B by multiplying the normal price of A with $x - \beta \cdot \varepsilon$, where β is the *exchange uncertainty discount factor*, which is characteristic for a given currency and may be mentioned in its description under the uncertainty discount standards. If the costs for a hypothetical exchange cannot be neglected one may multiply the resulting value with some factor $q \in (0, 1)$ close to one to account for the exchange costs.

Note that there are potentially different ways to estimate the normal price for a good or a service depending on whether the good is used directly for an economic purpose or is exchanged for another good in the market and depending on what good or service is used as a reference (i.e. good A in the above). Among all possible estimates one chooses the largest value as the normal price for a good or service under the currency. In particular, it is possible that a good or service, for which a normal price is mentioned explicitly in the description of the currency, may be considered to have an even higher normal price: This may occur for instance if this good or service can be exchanged for another good or service in the market which is also mentioned in the description and has an even higher normal price. In other words, the normal prices defined in the description are *minimal* normal prices to be precise.

Apart from the above there are additional corrections which may apply for the calculation of the value of the expected output of a proposed operation:

Output uncertainty: Once normal prices for every good or service from the expected output of an operation proposal have been estimated, one must take into account that the volume of the output might be subject to uncertainty as well. One may estimate the expected value of the output by multiplying the average normal price of the goods involved with the value $z - \gamma \cdot \delta$, where z is the expected volume, i.e. quantity, of the goods and services, δ is the standard deviation for the volume and where γ is the *output uncertainty discount factor*, which is characteristic for a given currency and may be mentioned in its description under the operation proposal evaluation standards.

Counter-party risk: Once the estimated value v of the output under the assumption that the operator fulfills his or her promises and obligations is obtained, one should also consider the possibility of a breach of contract. The most simple way to do so is to assume that the operator will either fulfill his or her obligations fully or take no steps whatsoever to conduct the operation. If the likelihood of the latter is κ then the expected value v is corrected to $v' = v - v \cdot \kappa \cdot \rho$, where ρ is the *counter-party risk discount factor*, which is characteristic for a given currency and may be mentioned in its description under the operation proposal evaluation standards.

SIDE EFFECTS: It is also theoretically possible that when a given operation is implemented a negative or positive effect on already ongoing operations occurs. In other words, it is possible that the expected output of other operations (which are already approved or about to be approved) or its expected value is diminished, which in turn would lower the overall value of existing liquidity in the respective currency. This may, for instance, be due to anticipated distortions in the price structure.
If such an effect is to be expected to occur to any meaningful extent, then the value of this proposal is to be corrected accordingly to reflect its actual net contribution to the respective currency.
It is also theoretically possible that a proposed operation, if implemented, would actually have a positive effect on the expected output of other operations, such that in the evaluation of this operation proposal a correction upwards is to be applied.

The above standards and definitions from the description of a currency allow to calculate the *overall estimated value of the economic output* from a proposed operation.

In the description of an operation proposal the applicant may also define deductions from the gross proceeds that would occur during or at the end of the operation depending on the situation. The actual *fair value of the operation* may then be calculated as the difference between the estimated value of the economic output and the overall expected deductions. Alternatively, one may first establish the *net economic output*, i.e. the output after all deductions, and then calculate the fair value of the operation as the *expected/estimated net economic output*, i.e. the net output (resp. its value) that is expected on average across all scenarios and under consideration of all deductions and risks.

B.2 Operation proposal

An operation proposal is written by the respective operator and should have the form and content specified in the description of the respective currency. It should contain the following information:

1. A description of the intended operation: what the liquidity will be exchanged for (i.e. the input), what the input will be used for (e.g. production that is expected to take place), what output it is expected to lead to and how the output will be marketed to obtain the actual proceeds (e.g. in terms of the chosen currency).

2. An enumeration of deductions from the gross proceeds the operator reserves the right to perform in order to cover internal costs and/or to make a profit. This includes a clear definition of the part of the gross proceeds that will actually be deleted from the system (deletable proceeds). Such deductions can be dependent on the gross proceeds or on other circumstances as long as a clear definition is provided. Note that the goods and services behind the deletable proceeds form the net economic output of the operation.

3. An estimated timetable for the different steps of the operation. This includes the expected duration of the operation as a whole, i.e. the expected time between the moment a proposal is granted and the expected maturity of the operation.

4. A description of the risks involved in the envisaged operation: What are the possible disruptions and potential consequence of such disruptions, as well as the likelihood or frequency of such disruptions based on the operator's experience or knowledge. Are there possibly any insurance or backup mechanisms to limit the damage in terms of the deletable proceeds?

5. The range in which the final deletable proceeds can be expected to be in and an explanation why it is believed that the requested liquidity amount is roughly the same as (and/or is not higher than) the estimated net value of the deletable proceeds (on average) under the

assumption that goods and services are sold at normal prices. This includes a declaration regarding which numbers were used for the normal prices and why.

6. The operator's own assessment of his or her experience, knowledge or expertise in conducting such operations, as well as relevant information regarding his or her credibility and willingness to fulfill his or her obligation to supply correct information and to conduct the operation as outlined in the description.

The operator must provide all relevant information regarding the above and cannot hide any important facts or circumstances.

The operator can be an individual or a company or a group of people etc. as long as in the proposal the operator, i.e. the obligated party, is properly identified.

B.3 Approval process

After an operation proposal was written the operator must attach the following information to the proposal:

1. An estimate for the time he or she believes is needed for a qualified neutral observer to assess the proposal and decide whether it should be accepted or not. This time is referred to as the processing time.

2. The fee that the operator is willing to pay to the referees if the proposal is accepted. This fee would be deleted from the requested amount and cannot exceed the latter. This fee cannot be lower than the minimal fee for operation proposals under this currency and cannot exceed the maximal fee.

After the operation proposal with the above information is submitted to the notaries and published by them, every referee can review the proposal. If no-one responds within the defined processing time the proposal is considered rejected. However, it is possible for every referee to write within that time a *supporting note* describing why the referee believes that the proposal is worthy of being accepted. For the note to be valid the referee must attach the referee stake as a security which he or she must be willing to loose in case the proposal does not get accepted. The supporting note is attached to the proposal. If no-one responds to this new situation within the processing time the proposal is granted, the referee receives his or her security back, the requested amount is created and provided to the operator after the fee was deducted and provided to the referee.
However, during the processing time after a supporting note was issued any of the remaining referees can write a rejecting note regarding the proposal and attach it to the supporting note. For this new note to be valid the second referee must also attach the referee stake as a security which he or she is willing to loose in case the proposal does get accepted. If no-one responds to this new situation, the second referee "wins", the proposal gets rejected, no new liquidity is created, the second referee receives his or her stake back, but also receives the stake from the first referee.
However, before the processing time runs out a third referee may write a supporting note contrary to the position of the second referee and so on.
The above procedure creates a *decision thread* consisting of alternating approvals and rejections. Every referee in the thread bets the referee stake as a security. The side which wins keeps their stakes, while the other side loses them. If the supporting side wins, the first referee obtains the fee, while the other referees who won obtain a referee stake. If the rejecting side wins, the supporting referees hand over a referee stake each to every of the rejecting referees.
It is in the interest of every referee who takes part in the process to make a strong argument in his or her note, in order to deter future referees from betting against his or her position. This may include information or arguments which have not been considered by previous referees. A referee may also simply point out mistakes in the line of argument of previous referees.

The referees must support or reject an operation proposal based on the following guidelines:

- An operation proposal should not be supported if it does not fulfill the requirements of this constitution or the description of the respective currency. Also, the processing time must be appropriately chosen and the fee must be within the bounds required in the description of the currency. For instance, a processing time which is lower than the estimated time a qualified referee needs to review such a proposal on average implies that the proposal must be rejected regardless of the content. The same applies to proposals which contain insufficient information to make a qualified decision based on the proposal together with the information publicly available at the time the proposal is submitted even if additional information becomes public during the review process.

- In order to support a proposal a referee must estimate the net value of the output described in the operation proposal under normal prices while applying evaluation and discount standards defined in the currency description. The referee can follow the calculations in the proposal and apply corrections whenever necessary. If his or her estimate is equal or is above the amount requested by the operator, i.e. the proposal is undervalued, the proposal can and should be supported. However, even if the amount estimated by the referee is below the requested amount, i.e. the proposal is somewhat overvalued, the proposal may still be supported if the difference is negligible or if such over-valuations are currently not more common or likely than similar under-valuations among granted proposals in this currency.

Referees can coordinate their activities outside the procedures defined in this document as long as it does not contradict any rules and principles described in this document and helps to improve the quality and efficiency of their work. For instance, referees can inform other referees that they are looking at a certain proposal at a given moment to avoid too many people studying the same proposal at the same time.

In general, referees should support rather than impede other referees in their work and should not benefit at the expense of operators or referees outside the procedures defined in this document.

Referees must prioritize the review of proposals recently supported by other referees over the evaluation of new proposals which no-one has supported yet. This prioritization is especially important when incorrect or questionable approvals are not rare or cannot be assumed to be unlikely, such that double-checking is necessary to keep the amount of "bad liquidity" created in the system to a minimum. In general, it is more important to prevent bad liquidity from being created than to create good liquidity of comparable amount.

Finally, the first referee of a decision thread, i.e. the one who issues the first supporting note, is referred to as the approving referee. There is a limit on how much liquidity a referee can (implicitly) create as approving referee. This limit is defined in the description of the respective currency. An initial supporting note for an operation proposal is invalid if the approving referee would exceed his or her limit in case the proposal is approved.

B.4 Referee appointments

In order to conduct reviews of operation proposals in a currency, a referee must be appointed to this function for the respective currency. For other currencies the process must be repeated.
A referee who wants to join a currency must create an application and publish the application via the notaries. After this the acting referees can review the application. A referee with power of appointment (i.e. any acting referee who has not appointed the maximal number of referees yet) can choose to support this application and appoint this individual as a new referee. However, this appointment is not approved yet. Furthermore, the appointing referee must attach the stake for the referee appointment process as defined in the description of the currency. He or she may loose this stake if the approval process fails. Now, other referees (not necessarily referees with appointment power) may write a rejecting note and attach it to the appointment note together with the stake and so on. In other words, the procedure is similar to the review process for operations. Supporting

and rejecting notes alternate and one of the sides wins when during the processing time (as defined in the description of the currency) after the last note no new notes are issued.
If the appointment is successful the appointing referee does not receive a fee. Instead he or she receives a part of all profits that this new referee will generate in his or her role as a referee for this currency during his or her tenure. This includes profits received from the referees appointed by this new referee in the future.
A referee application should contain the following information:

- A prospective referee should disclose relevant information on his or her past activities within the OCS, especially with respect to his or her activities as a referee in the past (if applicable).
- The application should contain information, declarations or other documents confirming that the referee is fully aware of the rules and guidelines he or she is expected to follow as a referee and that he or she intends to follow these rules and guidelines.
- The application must state the date at which the tenure of the new referee would start.

The acting referees must support or reject an application based on the following guidelines:

- A referee application must be rejected if the information listed above is missing or incomplete. An application must contain sufficient information to make a qualified decision based on the application together with the information publicly available at the time the application is submitted.
- A referee application cannot be rejected to restrict competition among referees. Rejections can be based on a lack of competence or trustworthiness.
- If the applicant is already an acting referee in this currency the starting date for the new tenure cannot be within the running tenure.
- The application must contain credible information confirming that the prospective referee is well-instructed, qualified and interested in following the rules and guidelines listed in this document and/or the description of the respective currency.

Similar to the approval process of operation proposals referees must prioritize the review of applications recently supported by other referees over the evaluation of new applications which no-one has supported yet. In general, it is more important to prevent a bad referee from being appointed than to appoint a new good referee.

When an application is approved the newly appointed referee receives a fixed amount of liquidity as defined in the description of the currency. This amount can be used to take part in decision threads. It can, however, only be used to this end and cannot be directly transferred to other participants.

B.5 Notarization process

All new entries must be notarized to be considered valid. An entry is for instance

- any supporting or rejecting note in the decision thread for a new operation proposal,
- any supporting or rejecting note in the decision thread for a referee application,
- a note of transfer of liquidity between operators.

An entry has usually at least one predecessor which serves as proof that the new entry is valid. However, this proof is insufficient as conflicting entries might already exist. The notaries check if there are any entries in circulation which would contradict the new entry. If there are

no contradictions an acting notary signs the new entry and sends it to another notary who signs the signed note etc. Once a certain amount of signatures of acting notaries is reached the entry is defined as notarized. Every signature has a time stamp, which stands for the time when the signature was added to the list. Every notary is appointed at some time and stops being a notary at a well known time in the future, which depends on when he or she was appointed. A notary signature is invalid if the time stamp is inconsistent with a notary's tenure. The date (and time) of a notarized entry is the time of the first time stamp in the sequence of signatures.

During the notarization process a notary who adds a signature must make sure that a large number of acting notaries becomes aware of this signature quickly (by sending it to them). If a notary can send the newly created signature to a small number of other acting notaries only, the signature should not be used and deleted instead.

A properly notarized entry is *fresh* for a limited period of time and stops being fresh at a well-known time which is a function of the entry's date. If an entry A is a predecessor of an entry B, than the date of B must be such that A is still fresh at that time, i.e. the difference between the time stamps of the two entries must be below the "freshness time". Otherwise the notarization of B is invalid per definition, while A is still valid but cannot become a predecessor of any new entry. This ensures that a freshly notarized entry can be used for new entries for a limited period of time only.

However, there is the possibility of *renotarization*, which occurs when a freshly notarized entry might run out of "freshness" and basically means that the notarized entry is itself notarized. These two notarized entries can be seen as equivalent to each other, however, one remains fresh for a longer period of time.

Renotarizations can occur arbitrarily often as long as the difference between the date of the previous notarized entry and the time stamp of the following is below the required freshness time.

Notaries are required to renotarize entries which theoretically still might be used as predecessors for one or more entries, but are not allowed to renotarize entries which cannot be used for new entries any longer. For instance, when liquidity is transferred from one operator to another, the entry which is the basis of the transfer cannot be used for other transfers to make sure that the same liquidity is not transferred twice to two different participants.

In general, the notarization of entries is to be automatized. I.e. a notary should not make the decisions regarding notarization him or herself, but let a software do it which may have been written by the notary or by somebody else, but which is employed and supervised by the notary on an infrastructure which is also supervised by him or her. In other words, the notary should feel responsible for making sure that the software and hardware is fulfilling its purpose correctly at all times. The software places signatures on a notary's behalf based on information contained in the entry and based on information, i.e. other entries, primarily the notarized ones, obtained from other sources. Those other sources include other notaries first and foremost, which means that the software run by a notary should exchange entries with the software run by other notaries to make sure that everybody is up to date.

A notary should be aware of all other acting notaries and be able to exchange information with them and, in addition, provide information, which is needed for connecting to other acting notaries, to the public.

A notary should store as many of the notarized entries as possible, especially the more recently notarized entries. The most recent entries should be all known to the notary and be stored on his or her devices, ready to be accessed. Furthermore, the notary is required to provide this information to other notaries upon request or when the information is recent and needs to be actively disseminated. Formally correct and properly notarized entries are stored by the notary in a database, which must be open to all members of the community such that the data is relatively easy and convenient to access.

B.6 Appointment of notaries

A notary lineage is identified by its description, which contains the following information:

- The maximal number N of acting notaries.
- The tenure T of every notary.
- The public keys of the initial N notaries.
- The time t when the first of the N initial notaries becomes an acting notary.
- The number M of signatures (from different notaries) necessary to perform a notarization (this number should not exceed N).
- The maximal number K of notaries who are appointed but (simultaneously) waiting to become acting notaries.
- The maximal notarization time U, which is the maximal time difference between two notary signatures in the same notarizing block (not necessarily two consecutive).
- The freshness time S of a freshly notarized entry.
- The notarization fee (fee withheld in return for a successful notarization) as a percentage of the liquidity amount contained in an entry.
- The penalty factor F_1 for lost decision threads regarding new notary appointments. F_1 is not below zero and is not larger than 1.
- The penalty factor F_2 for ignored decision threads regarding new notary appointments. F_2 should be between F_1 and 1.
- The time R which is regarded as sufficient to review a notary application.

The notaries are counted based on the order in which they were appointed. Every appointment is an entry which points to the previous appointment, except for notaries which are mentioned in the description of the lineage. The number of the first notary is 1. If k is the number of a notary he or she becomes an acting notary at time $t + (k-1) \cdot T/N$ by definition and stops being a notary at time $t + (k-1) \cdot T/N + T$ which is also the time when the notary with number $k + N$ becomes an acting notary. Thus, notaries constantly come and go.

Acting notaries can perform notarizations but must also appoint future notaries. A new notary may be appointed whenever there are less than K notaries who were already appointed but are still waiting to become acting notaries. This restriction makes sure that there are not too many notaries appointed in advance at any moment in time. The appointment process consists of the following steps:

- The prospective notary writes an application and sends it to the acting notaries.
- The notaries may ignore the application, but every acting notary may issue within time R a supporting note and attach it to the application.
- If no other action occurs within time R the new notary is appointed. However, other notaries may issue a rejecting note and so on. This leads to a decision thread which terminates whenever there is no change for time R. If the last note was supportive the termination time of the thread is the time the new notary appointment may be finalized. The new notary's number is determined by the number of appointments before that time.

Those participants of the decision thread who lost receive a penalty: They hand over the part $1-F_1$ of all the fees collected or to be collected during their respective tenures to those who won and keep only the part F_1. Note that a notary receives the fees he or she collected only at the time the notary is not an acting notary any more. If such a losing notary looses another thread then the part $1-F_1^2 = (1-F_1) + F_1 \cdot (1-F_1)$ of all fees collected is handed over in total to other notaries as a consequence.
Those who abstain from a vote, which led to an approval, hand over $1-F_2$ to those who won in order to discourage not taking part in decision threads. Thus, not taking part is also a loss in a sense. This, however, does not apply if the notary application was not approved, i.e. the last entry of the thread is a rejection. Neither does the penalty apply if the notary who abstained from the vote is not an acting notary at the time the thread terminates.

Note that the set of eligible participants in a notary application decision thread is limited to those notaries who were already acting notaries at the time the application was submitted. Also, there is no profit or loss for a participating notary if he or she is not acting at the time the thread is terminated.

A notary application should contain the following information:

- The prospective notary should disclose relevant information regarding his or her past activities within the OCS, especially with respect to his or her activities as a notary in the past (if applicable).
- The application should contain credible information, declarations or other documents confirming that the notary is fully aware of the rules and guidelines he or she is expected to follow as a notary and that he or she intends and is likely to follow these rules and guidelines.

The acting notaries must support or reject an application based on the following guidelines:

- It is highly desirable for the application to fulfill the above formal requirements.
- It is highly desirable for a prospective notary to have the software, the hardware and the technical expertise to serve as a notary and to be able to start doing so immediately when the tenure begins.
- A notary does not have to be an actual person, but can also be an organization or a team of people. Furthermore, two or more different notaries can be the same person, but the person has to use different key pairs for each notary. In general, a key pair should be used only during a specific tenure for a specific notary lineage and should not be used anywhere else.
- It is desirable for notaries to be as unrelated as possible: Ideally different people who are located in different countries, or running software on different computers using different internet connections. In other words, when considering two applications from two equally trustworthy and reliable people the one who is more unrelated to existing notaries is more desirable.
- An application should be rejected if there are more suitable applications which can also be approved immediately or are likely to appear soon enough.
- In general new notaries should be appointed sooner than later to make sure that there is insignificant risk of the lineage being interrupted. If no good or insufficient applications are available acting notaries should apply themselves (possibly multiple times).

Similar to the approval process of operation proposals by referees

- notaries must prioritize the review of applications recently supported by other notaries over the evaluation of new applications which no-one has supported yet. In general, it is more important to prevent a bad notary from being appointed than to appoint a new good notary unless there is exceptional time pressure.

- Notaries can coordinate their activities outside the procedures defined in this document as long as it does not contradict any rules and principles described in this document and helps to improve the quality and efficiency of their service.

In general, notaries should support rather than impede other notaries in their service for the community and cannot benefit at the expense of operators or referees or other notaries outside the procedures defined in this document.

When a decision thread regarding a new notary appointment successfully terminates and the supporting side wins, the designated notary accepts the appointment by sending a new entry in the lineage for notarization. I.e. a new entry is created which points to the entry according to which the previous notary in the lineage was appointed and this new entry is sent to acting notaries for notarization. Acting notaries must notarize the entry if the respective decision thread indeed led to an approval. Otherwise the entry should not be notarized. It is the notarized entry in the lineage which serves as necessary and sufficient proof to the non-notaries that this new notary was in fact appointed. Also, the position in the lineage, i.e. the distance to the first notary of the lineage allows to reconstruct the times when the designated notary begins and stops to be an acting notary.

B.7 Compensation of notaries

As described in the previous section the basis of the compensation of a notary is the notarization fee. This fee is charged for entries which are used to transfer liquidity and are successfully notarized. The fee is received by the notary who started the notarization of the respective entry by placing the first signature. However, some of the total fees collected during the tenure of a notary must be handed over to other notaries due to lost or ignored decision threads regarding new notary appointments.

A notary cannot use any of the liquidity collected during his or her tenure before the tenure has ended. However, a notary can claim a share of the fees collected by another notary as described above beginning with the time the tenure of this other notary ends and his or her share distribution is finalized even if the notary's own tenure has not ended yet.

The overall share of collected fees that a notary hands over to winners of decision threads that occurred during his or her tenure is divided between all eligible winners, with eligibility defined in the previous subsection. The share that an eligible winner receives depends on how often this notary "lost to" the other notary (eligible winner) and according to what scenario. More precisely, the share that the other notary receives is proportional to $(1 - F_1) \cdot W_1 + (1 - F_2) \cdot W_2$, where W_1 is the number of wins (of the other notary) in decision threads that both notaries took part in (and this notary lost) and W_2 the number of wins (of the other notary) in decision threads which lead to approvals and that this notary has ignored.

C Technical Implementation

On the technical level the complete state of the OCS is determined by a finite set of entries. Every entry is a finite sequence of bytes. It begins with a *head byte* and ends with the EOF byte, which is the byte 0x04 (hexadecimal system) or 00000100 (binary system). The head byte determines the type of the entry, i.e. the meaning or role of the entry within the system. There are 15 different types, which are:

Type	Head Byte	Meaning
1	0x20	Description of a notary lineage. I.e. the whole notary lineage description is one entry.
2	0x21	An entry in a notary lineage which confirms that a new notary was appointed.
3	0x22	A notary application.
4	0x23	A supporting or a rejecting note in the notary application review process, i.e. an entry in the decision thread.
5	0x24	Description of a currency.
6	0x25	Application of a new referee.
7	0x26	A new operation proposal.
8	0x27	A supporting or a rejecting note in a referee application or operation proposal review process, i.e. an entry in the decision thread.
9	0x28	An entry marking the end of a notary application or referee application or operation proposal review process.
10	0x29	A note regarding liquidity transfer between participants.
11	0x2A	An entry containing a new public key as a request to register it.
12	0x2B	A signed entry for any entry of the above types except type 1. This implies that the signature is placed by a non-notary (or a notary but not for the purpose of notarization).
13	0x2C	A signed entry for an entry of type 12. This implies that the signature is placed by a notary (for the purpose of notarization or renotarization).
14	0x2D	A liquidity claim for a given participant and currency/obligation.
15	0x2E	Description of an obligation.

C.1 Type 11 - Public key

For signatures under type 12 and 13 public-key cryptography must be used. In other words, every participant generates a key pair: While the private key remains known to the respective owner only, the public key is put into an entry of type 11 and sent to the notaries. The private key is used to make signatures, the public key is used to verify them. The length of the public key in bytes is variable as is the cryptographic algorithm used to sign documents and verify signatures. The public key as a sequence of bytes must contain sufficient information to clearly identify (based on the context) the cryptographic method used and how to interpret the key. The chosen method must be open source, fairly standard and well-known, reasonably inexpensive to use and reasonably safe.

The structure of an entry of type 11 is the following:

1. The first byte is 0x2A.
2. The next 8 bytes are an unsigned long integer defining the prospective date until which the key is valid. The natural number given by the long integer is interpreted as a Unix epoch time stamp in milliseconds.
3. The following 8 bytes are an unsigned long integer defining the length of part 4. (in bytes) below:
4. What follows is the public key as a byte sequence.
5. The last byte is 0x04.

A new entry of type 11 is send to the notaries for notarization. By signing the entry the respective notary confirms that the public key is formally correct, which implies in particular that the meaning

of the key is clear and unambiguous, as is the underlying cryptographic algorithm, and the assumed algorithm satisfies the above criteria.

If the above requirements are not fulfilled the type 11 entry is formally incorrect and is not subject to notarization.

It is possible to revoke a public key by issuing a new type 11 entry with the same key but with a validity date which is in the past (e.g. 0). In general, if there are different properly signed (see section C.3) and notarized (see section C.2) type 11 entries the validity date of the public key is the minimum of all the validity dates mentioned in these entries.

A participant, i.e. an operator, a referee or a notary, is identified by a public key and can be a person or an organization/team of people. It is essentially everybody who holds and is supposed to hold the associated private key or to sign using it.

Once a type 11 entry becomes notarized it is not subject to renotarization, i.e. it remains fresh for ever by definition.

C.2 Type 13 - Notary signature

An entry of type 13 plays the role of a signature "placed beneath" an entry of type 12. Let S be the entry that needs to be signed as a sequence of bytes, but without the last byte 0x04. A signed entry of type 13 for this entry S has the following structure:

1. The first byte is 0x2C.

2. The next 4 bytes are an unsigned integer denoting the number k of the notary within the lineage. The smallest possible number for k is 1.

3. The next 8 bytes are an unsigned long integer defining the ID for this signed entry as chosen by the notary. The notary must make sure that this ID is chosen only once, i.e. for this signed entry only.

4. The next 8 bytes are an unsigned long integer defining the time when this entry is generated as a Unix epoch time stamp in milliseconds. The unsigned integer k, together with the above ID and this time stamp form the *complete ID*. The complete ID is used to identify this entry and to distinguish it from all other entries of type 13 signed by notaries of this lineage.

5. The next 20 bytes denote the complete ID of the previous type 13 entry the current type 13 entry is pointing to. If this is the very first entry in a notarization (excluding renotarizations) these 20 bytes are identical to the previous 20 bytes above (i.e. the complete ID of this entry). Otherwise, let T be this previous type 13 entry as a sequence of bytes, but without the last byte 0x04.

6. The next 20 bytes denote the complete ID of the first type 13 entry in the sequence of entries this entry is part of. If this is the first entry in a notarization (or renotarization) these 20 bytes are identical to the complete ID of this entry.

   For the following let R be the concatenation of the following five sequences: the complete ID of this entry, the complete ID under 5. (preceding entry), the complete ID under 6. (first entry), S and T (in this order). We set R as the concatenation of the first four objects in case T is not defined.

7. The following 8 bytes are an unsigned long integer defining the length of part 8. (in bytes) below:

8. The following bytes are a cryptographically signed version of R signed using the private key of the notary.

9. The final step is to add the byte 0x04.

Note that since the number k of the notary is encoded in an entry of type 13 the public key of the notary can be reconstructed either from the description of the notary lineage or from one of the following lineage entries. This public key can then be used to verify the signature in the respective entry of type 13. Furthermore, if this is the first type 13 entry for the respective type 12 entry, the signed version of R must be such that R itself can be reconstructed from this signed version using the public key of the notary. Note that in this case S is obtained from R by removing the first 60 bytes.

Also, note that only type 13 entries have IDs and can point to each other. All other entries can be identified only by including the actual entry (or its hash value) in the respective referencing entry. However, once an entry has been "wrapped" into a type 13 entry by an acting notary it can be implicitly referenced using the respective complete ID. In order to sign the same entry a second time, the second notary creates a new type 13 for the same entry S but includes the complete ID of the first type 13 entry under 5. and 6. above. By doing so the second notary confirms that the initial entry and the existing type 13 entry are formally correct.
After repeating this $M-2$ more times and collecting M correct signatures in total, the underlying entry is defined as notarized and is identified by the complete ID of the first signature in the sequence.

This process of signature collection is moderated by the initial notary: After creating the initial type 13 entry the respective notary sends it to all other notaries and appoints the next notary to sign. Upon receiving the next signature the moderating notary distributes it among all notaries and appoints the third notary etc. The notarization process must be aborted whenever the connection to a large number of other notaries is interrupted and there is the possibility of the notarization of conflicting entries due to a temporary split in the notary community. Furthermore, not only the moderating notary is responsible for making sure that everyone is aware of a notarization going on, but also each of the other participating notaries.

A renotarization is analogous to a first notarization. In the first entry of the renotarization, however, one does not reference the last entry of the preceding notarization, but references the first entry of the preceding notarization (or renotarization) under 5. instead, while placing a copy of the complete ID of this first entry (of the renotarization) under 6. This is the only situation when the complete ID under 6. can be "later" then the ID under 5. After a successful renotarization the now renotarized entry may be (also) identified by the complete ID of the first signature in the renotarization.

Note that within a notarizing or renotarizing block of M type 13 entries all the signatures must be from different notaries.

Note also that there is a before-after-relationship between type 13 entries: Firstly, entries are compared based on their time stamps. If the time stamps are the same, the "earlier" entry is the one with a smaller notary number. And if the notary numbers are the same than the "earlier" entry is the one with a smaller ID if interpreted as an unsigned long integer.

In order to sign a type 12 entry and produce a type 13 entry, the respective notary must first verify that the respective type 12 entry is formally correct and that nothing speaks against it. Also, for the resulting type 13 entry to be formally correct, the notary must be an acting notary at the time of signing, which can be verified based on the number k and the description of the notary lineage. Furthermore, the signature must be valid, which can be verified using the public key of the notary with number k. This key is contained either in a type 1 or type 2 entry.

If we have a type 13 signature which is not the first in a notarization, the notary must also verify that all the preceding type 13 entries are formally correct. Furthermore, the maximal notarization time U between signatures must be respected. I.e. if the first signature in the current notarizing or renotarizing block was placed more than (or exactly) the time U in the past then no following signatures can be placed and if a follow-up signature is placed nevertheless then the resulting block

of type 13 entries is formally incorrect. Also, a type 13 entry is formally incorrect if the before-after-relationship (implied by the IDs) with the previous type 13 entries in the chain is violated.

A given type 12 entry (as a sequence of bytes) should have only one initial notarization and every notarization or renotarization should have at most one subsequent renotarization.

A new notarization or renotarization should not be initiated if the respective moderating notary is aware of an ongoing notarization of a conflicting entry. It is, however, still possible that two different notarizations of conflicting underlying entries (or the same underlying entry but different initial type 13 entries) are started around the same time, such that notaries are confronted with both being in notarization. If either one of the entries is formally incorrect (even if the other entry is ignored) or cannot be notarized any more due to time-out, then the notarization of this entry should obviously be aborted without affecting the notarization of the other entry. If both entries could be notarized, assuming the other entry would not exist, then the notarization of both entries should be aborted (assuming they in fact exclude each other).

For renotarizations the following must be considered: There is an order in which notaries attempt to produce a follow up notarization entry. The list of eligible moderators consists of all future notaries who would be acting notaries at the half-time between the time stamp of the current latest preceding notarization entry and the time when this entry stops to be fresh. This half-time is the earliest renotarization time for this preceding entry. When this particular time is reached the first notary in the list creates and signs a type 13 entry starting a renotarization process. If a notarization does not occur within the maximal notarization time (for this lineage) after the earliest notarization time, then the attempt has failed by definition and the next notary in the list must attempt to create and notarize a type 13 entry and so on. Each notary has a time window which is in size equal to the maximal notarization time (for this sub-lineage) and these windows are disjoint.

The above procedure for renotarizations applies only if a successor entry is produced before the respective sub-lineage is "cut off" (according to the definition of the type 1 entry). Otherwise, an initial type 13 entry must be produced by the first notary of a follow-up sub-lineage and notarized following normal time constraints and procedures.

C.3 Type 12 - Simple signature

Any entry of type 2 to 11 can be signed by a non-notary member such that an entry of type 12 is produced. Let S be the entry that needs to be signed as a sequence of bytes, but without the last byte 0x04. A signed entry of type 12 for this entry S has the following structure:

1. The first byte is 0x2B.
2. The following 8 bytes are an unsigned long integer defining the length of part 3. (in bytes) below:
3. The following bytes are S.
4. If S is an entry of type 11 this part 4. is completely omitted. Otherwise this part has a length of 20 bytes and is the complete ID of a notarized version of an entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which in turn is wrapping an entry of type 11.
5. The next 4 bytes are an unsigned integer denoting the number k of the notary who must be the first to sign this entry during notarization.
6. The next 8 bytes are an unsigned long integer defining the time t until which this entry is to be notarized as a Unix epoch time stamp in milliseconds.
7. The following 8 bytes are an unsigned long integer defining the length of part 8. (in bytes) below:

8. The following bytes are a cryptographically signed version of the byte sequence consisting of S, immediately followed by the 20 bytes of part 4. (if applicable) and then k and t (12 additional bytes in total). I.e. this sequence is signed using the private key of the respective member and the resulting byte sequence is placed here.

9. The final step is to add the byte 0x04.

Usually the first entry that is signed by a new member according to the above is an entry of type 11 which contains the public key. Note that an entry of type 11 is not sent to the notaries in plain form, but first signed and wrapped into an entry of type 12 as described above. This confirms that the submission of an entry of type 11 is in fact approved by the owner of the key pair.

Regardless of whether the signed entry S is of type 11 or not the public key of the signatory can be reconstructed as there is at least a pointer to an entry of type 11. Using this public key the validity of the signature can be verified.

For a type 12 entry to be formally correct the underlying entry must be formally correct. Furthermore, the signature must be correct and the associated public key must be valid at the time the notary receives the entry (or a previously signed version of it) for signing. Otherwise the entry is not subject to notarization.

Also, for a type 13 entry based on a type 12 entry to be formally correct the first signature in the list of notary signatures must be placed by the notary with the number k defined in the type 12 entry. Furthermore, all signatures must be placed before the time limit specified in the type 12 entry. This restriction, however, does not apply to renotarization entries if they are based on a formally correct notarized entry.

If notarization fails, i.e. a formally correct type 13 notarized entry based on a given type 12 entry is not produced until the specified time, for whatever reason, a new type 12 entry has to be produced and submitted (preferably to a different notary) by the owner of the key pair.

C.4 Type 1 - Notary lineage description

The description of a notary lineage is the only entry which is not subject to signing or notarization. Like any other entry it is a sequence of bytes. For a basic type 1 entry the sequence has the following form:

1. The first byte is 0x20.

2. The following two bytes are an unsigned short integer denoting the maximal number N of acting notaries.

3. The following two bytes are an unsigned short integer denoting the number M of notary signatures to perform a notarization.

4. The following two bytes are an unsigned short integer denoting the maximal number K of notaries who are appointed but not yet acting notaries.

5. The following $4 \cdot N$ bytes are unsigned integers denoting the lengths in bytes of the public keys of the first N notaries of the lineage.

6. The following bytes are the public keys of the first N notaries of the lineage. Note that the public key byte sequences follow directly one after another without a delimiter, as the length of every key is previously defined. In particular it is known in advance when the last public key ends.

7. The next 4 bytes are an unsigned integer defining the tenure of every notary in seconds.

8. The next 4 bytes are an unsigned integer defining the maximal notarization time in milliseconds.

9. The next 4 bytes are an unsigned integer defining the freshness time of a notarized entry in seconds.

10. The next 4 bytes are an unsigned integer defining the time R in seconds regarded as sufficient to review a notary application.

11. The next 8 bytes are an unsigned long integer defining the begin of the tenure of the first notary as a Unix epoch time stamp in milliseconds.

12. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the percental notarization fee as a number between 0 and 100.

13. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the penalty factor F_1.

14. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the penalty factor F_2.

15. The last byte is 0x04. Note that even without this EOF byte it is known in advance when the entry ends due to the structure of the previous parts of the entry as defined above.

We refer to an entry with the above structure as a *basic type* 1 *entry*. A general type 1 entry is in some sense a list of basic type 1 entries, where the last lineage in the list is the latest lineage the parameters of which are valid for the time being. More precisely, the list has the following form:

1. The first byte is 0x20.

2. What follows is a basic type 1 entry without the first byte, which is 0x20, and without the last byte, which is 0x04.

3. The next 8 bytes are an unsigned long integer defining the last time at which an entry properly notarized under the above notary lineage is recognized as notarized by the latest notary lineage. We refer to this as the *cut-off time* of the lineage defined under 2.

4. The next 8 bytes are an unsigned long integer defining the *extended freshness time* (in milliseconds), which is the time for which every entry which was a fresh notarized entry under the above notary lineage at the cut-off time defined above remains a fresh notarized entry beyond this moment by definition.

5. The next 8 bytes are an unsigned long integer defining the first time at which an entry properly notarized under the now following notary lineage is recognized as notarized by the latest notary lineage. We refer to this as the *kick-off time* of the now following lineage.

6. What follows are arbitrarily many repetitions of steps 2., 3., 4. and 5. At the end, however, there is only a step 2. without steps 3., 4. and 5.

7. The last byte is 0x04.

The times above are Unix epoch time stamps in milliseconds.

For a generalized type 1 entry to be consistent the kick-off time of a notary sub-lineage must be larger than the cut-off time of the previous notary sub-lineage. Also, the cut-off time of a notary lineage cannot be smaller than its kick-off time.

Such a generalized type 1 entry can be used to fix corrupted notary lineages: Just create a new lineage under which entries from the previous are still recognized to some extent. Remark that

- By convention, every notarized entry from any of the lineages in the list which is not the latest lineage must be signed by the first notary of the latest sub-lineage to be considered valid under the new lineage. To this end a properly signed type 13 entry (called confirmation entry) must be produced by the first notary which canonically extends the existing sequence of signatures for the respective notarization entry. In particular, it has as predecessor ID the complete ID of the last signature in the respective preexisting notarizing sequence. The complete ID of the confirmation entry itself, however, is independent of the underlying entry: It is encoded by the number 1 for the notary number of the first notary, the number 0 for the ID and a time stamp which is always equal to the maximum of the kick-off time for the new sub-lineage and the tenure start for its first notary.

- Because of the above constraints for the kick-off and the cut-off times one can reconstruct based on the time stamp of a notarized entry which of the sub-lineages in the list it belongs to.

- Note the difference between the notions *notary number* and the *total notary number*. The latter is a pair consisting of the notary number and the number of the sub-lineage the notary belongs to. Using the notary number alone makes sense only if the sub-lineage is clear from the context.

C.5 Type 2 - Notary lineage entry

A type 2 entry has the following structure:

1. The first byte is 0x21.

2. The following 20 bytes are the complete ID of the notarized entry of type 2 this entry is pointing to. Technically, this is the complete ID of an entry of type 13 which is wrapping an entry of type 12, which is wrapping an entry of type 2, which is the predecessor of this entry in the respective notary lineage. If this entry is the first entry of type 2 in the lineage, then all 20 bytes are set to 0x00. If this is the first entry in a sub-lineage of a general type 1 entry then the complete ID of the last type 2 entry of the previous sub-lineage is placed here (or 0x00 if no previous type 2 entry exists).

3. The next 20 bytes denote the complete ID of the notarized version of the entry of type 9 with which the notary application decision thread, which justifies this appointment, has ended. Technically, it is the complete ID of a type 13 entry which is a notarized version of a type 12 entry, which in turn is wrapping a type 9 entry.

4. The following 4 bytes are an unsigned integer denoting the length of the public key of the newly appointed notary in bytes.

5. The following bytes are the public key (as a byte sequence) of the newly appointed notary.

6. The last byte is 0x04.

A type 2 entry must be signed by the respective designated notary and the resulting type 12 entry is sent to the notaries for notarization and dissemination.

For a type 2 entry to be formally correct, the respective public key should not appear anywhere else in the lineage. Furthermore, the two referenced entries (type 2, if applicable, and type 9) must be formally correct and fresh notarized entries. Finally, the identified decision thread must be for the notary with the same public key and must have resulted in an approval and must be formally correct and recent. "Recent" means that the thread must have terminated not more the one notary tenure in the past. "Formally correct" means that all entries of the decision thread are formally correct, which is equivalent to the last (notarized) entry being formally correct.

C.6 Type 3 - Notary application

A type 3 entry has the following structure:

1. The first byte is 0x22.
2. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the applicant.
3. The following 8 bytes are an unsigned long integer defining the length of part 4. (in bytes) below:
4. The following part is the actual application. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.
5. The last byte is 0x04.

A type 3 entry must be signed by the respective applicant and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct, it must be signed with the same key as the one contained in the underlying type 3 entry.

Apart from having the above structure a type 3 entry must be pointing to a formally correct and fresh type 13 notarized entry to be formally correct as well. Also, the specified public key must be such that there were and are no acting notaries identified by this key.

C.7 Type 4 - Entry in a notary application decision thread

A type 4 entry has the following structure:

1. The first byte is 0x23.
2. The next 4 bytes are an unsigned integer denoting the number k within the lineage of the notary, who issues this supporting or rejection note.
3. The next 20 bytes denote the complete ID of the notarized version of the application or the previous entry in the decision thread. Technically, it is the complete ID of a type 13 entry which is a notarized version of a type 12 entry, which is wrapping a type 3 or type 4 entry.
4. The following 8 bytes are an unsigned long integer defining the length of part 5. (in bytes) below:
5. The following part is the actual supporting or rejection note. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.
6. The last byte is 0x04.

Whether a note is supporting or rejecting is determined by the position of this entry in the decision thread.

A type 4 entry must be signed by the respective acting notary and the resulting type 12 entry is sent to other notaries for notarization and dissemination. For the type 12 entry to be formally correct, it must be signed with the key of the notary with the number k.

A notary is not allowed to place an initial approving note to a notary application if there are already N pending notary application decision threads with the initial note having been placed by the same refereeing notary. Here N is the maximal number of acting notaries (for the respective (sub-)lineage).

Apart from having the above structure a type 4 entry must be pointing to a formally correct and fresh type 13 notarized entry to be formally correct as well. In addition, when the type 4 entry is provided to the notaries for notarization the processing time (beginning with the time stamp of the mentioned type 13 entry) cannot have elapsed. Finally, the type 4 entry is formally incorrect if there is already another entry at this place of the decision thread or if the notary already took part in this decision thread or if the notary is not an acting notary any more or if this notary was not an acting notary at the time the notary application was submitted.

C.8 Type 5 - Currency description

A type 5 entry has the following structure:

1. The first byte is 0x24.
2. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the first referee.
3. The next 4 bytes are an unsigned integer defining the tenure of every referee in seconds.
4. The next 4 bytes are an unsigned integer defining the processing time (for one step) in the referee approval process in seconds.
5. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the referee stake in the referee approval process.
6. The following two bytes are an unsigned short integer denoting the maximal number of new referees an acting referee can appoint.
7. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity provided to every newly appointed referee.
8. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the maximal total amount of liquidity that can be created by one referee via approval of operation proposals during his or her tenure.
9. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the percentage of all profits of a referee that must go to the referee who appointed him or her (a number between 0 and 100).
10. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the minimal amount of liquidity to be transferred in standard liquidity transfers.
11. The next 4 bytes are an unsigned integer defining the minimal processing time for an operation proposal in seconds.
12. The next 4 bytes are an unsigned integer defining the maximal processing time for an operation proposal in seconds.
13. The following two bytes are an unsigned short integer denoting the number m of possible requested amounts considered below.
14. The following $32 \cdot m$ bytes are m quadruples of IEEE 754-1985 binary64 floating-point numbers, where the first element of the quadruple denotes a possible requested amount, the second the associated minimal fee, the third the associated maximal fee (which cannot exceed the requested amount) and the last the associated referee stake based on the requested amount.

This implicitly defines the minimal and maximal fees (for all requested amounts) using linear interpolation and extrapolation. Also, this defines the referee stake based on the requested amount again using linear interpolation and extrapolation. The final referee stake is the maximum of the referee stake based on the requested amount and the referee stake based on the fee, which is defined below:

15. The following two bytes are an unsigned short integer denoting the number l of possible fee amounts considered below.

16. The following $16 \cdot l$ bytes are l pairs of IEEE 754-1985 binary64 floating-point numbers, where the first denotes a possible operation proposal fee and the second the associated referee stake based on the fee. The referee stakes based on the fee for all other possible fees is defined implicitly using linear interpolation and extrapolation.

17. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the basic rate at which all outstanding liquidity is deleted (annual discount rate, e.g. -5 standing for 5% discount per year).

18. The following 8 bytes are an unsigned long integer defining the length of part 18. (in bytes) below:

19. The following part is the actual description of the currency and contains all the required information and guidelines not defined above. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.

20. The last byte is 0x04.

A type 5 entry must be signed by the respective first referee and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct, it must be signed with the same key as the one mentioned under 2. in the underlying type 5 entry.

In order to be formally correct (from the perspective of a notary lineage) a type 5 entry must have the above structure.

Also, the above numbers must be formally correct in light of section B.1.

Type 5, type 15 and type 11 entries are different from all other entries in that they do not have to be renotarized. I.e. once notarized they remain fresh for ever by definition.

C.9 Type 6 - Referee application

A type 6 entry has the following structure:

1. The first byte is 0x25.

2. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the applicant.

3. The following 20 bytes are the complete ID of a notarized entry of type 5. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 5, which contains the description of the respective currency.

4. The next 8 bytes are an unsigned long integer defining the begin of the tenure of this referee as a Unix epoch time stamp in milliseconds.

5. The following 8 bytes are an unsigned long integer defining the length of part 6. (in bytes) below:

6. The following part is the actual application. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.

7. The last byte is 0x04.

A type 6 entry must be signed by the respective applicant and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct, it must be signed with the same key as the one contained in the underlying type 6 entry.

Apart from having the above structure a type 6 entry must be pointing to formally correct and fresh type 13 notarized entries to be formally correct as well. In addition, the begin of tenure cannot be more than one referee tenure in the future. Also, only one pending application per referee and currency is allowed and tenures (for the same referee and currency) may not intersect.

C.10 Type 7 - Operation proposal

A type 7 entry has the following structure:

1. The first byte is 0x26.

2. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the applicant.

3. The following 20 bytes are the complete ID of a notarized entry of type 5. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 5, which contains the description of the respective currency.

4. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity requested by the operator for this operation.

5. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity the operator is willing to pay as a fee if the proposal is granted.

6. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity the operator is agreeing to delete in case the proposal is not granted.

7. This part is completely omitted if the number under 6. is 0. Otherwise this part consists of 20 bytes denoting the complete ID of the notarized version of a type 14 entry. Technically, it is the complete ID of a type 13 entry which is a notarized version of a type 12 entry, which is wrapping a type 14 entry. This entry must be free and carry sufficient liquidity to cover the liquidity the operator has chosen under 6. above. Also, the holding participant under this type 14 entry must be the applicant in this operation proposal.

8. The next 4 bytes are an unsigned integer defining the processing time for this proposal in seconds.

9. The following 8 bytes are an unsigned long integer defining the length of part 10. (in bytes) below:

10. The following part is the actual operation proposal and contains all the information of the proposal not mentioned above. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.

11. The last byte is 0x04.

A type 7 entry must be signed by the respective applicant and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct, it must be signed with the same key as the one contained in the underlying type 7 entry.

Apart from having the above structure a type 7 entry must be pointing to formally correct and fresh type 13 notarized entries to be formally correct as well. Also, all the numbers must be within the bounds specified in the description of the respective currency.

C.11 Type 8 - Entry in an operation proposal or referee application decision thread

A type 8 entry has the following structure:

1. The first byte is 0x27.
2. The next 20 bytes denote the complete ID of the notarized version of the entry of type 9 according to which the supporting or rejecting referee was appointed as a referee. I.e. it is the final entry in the respective decision thread. Technically, it is the complete ID of a type 13 entry which is a notarized version of a type 12 entry, which in turn is wrapping a type 9 entry. If this is the first referee for the currency, the complete ID of the respective type 5 entry is provided instead.
3. The next 20 bytes denote the complete ID of the notarized version of the operation proposal / referee application or the previous entry in the decision thread. Technically, it is the complete ID of a type 13 entry which is a notarized version of a type 12 entry, which is wrapping a type 6 or type 7 or type 8 entry.
4. The next 20 bytes denote the complete ID of the notarized version of a type 14 entry. Technically, it is the complete ID of a type 13 entry which is a notarized version of a type 12 entry, which is wrapping a type 14 entry. This entry must be free and carry sufficient liquidity to cover the referee stake for this operation proposal or referee application review process. Also, the holding participant under this type 14 entry must be the supporting or rejecting referee.
5. The following 8 bytes are an unsigned long integer defining the length of part 6. (in bytes) below:
6. The following part is the actual supporting or rejection note. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.
7. The last byte is 0x04.

Whether a note is supporting or rejecting is determined by the position of this entry in the decision thread.

A type 8 entry must be signed by the respective acting referee and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct it must be signed with the public key of the supporting/rejecting referee.

Apart from having the above structure a type 8 entry must be pointing to formally correct and fresh type 13 notarized entries to be formally correct as well. In addition, when the type 8 entry is provided to the notaries for notarization the corresponding processing time (beginning with the time stamp of the preceding entry) cannot have elapsed. Also, the signing referee must be an acting referee, which can be checked using the description of the respective currency and the information from the decision thread according to which the signing referee was appointed. Also, if this is the first supporting note in the decision thread, the referee must be within his or her limit for total liquidity creation (in case we have an operation proposal) or his or her limit for appointed referees

(in case we have a referee application) as defined in the description of the currency. Finally, the entry is formally incorrect if there is already another entry at this step of the decision thread or if the referee already took part in the decision thread or if the referee is the applicant.

C.12 Type 9 - End of a decision thread

A type 9 entry has the following structure:

1. The first byte is 0x28.
2. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the participant who will sign this entry.
3. The next 20 bytes denote the complete ID of the notarized version of the operation proposal / referee application / notary application or the last entry in the decision thread. Technically, it is the complete ID of a type 13 entry which is any notarized version of a type 12 entry, which is wrapping a type 3, 4, 6, 7 or a type 8 entry, which is the final entry in the respective decision thread.
4. The last byte is 0x04.

A type 9 entry must be signed by the respective participant and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct it must be signed with the specified key. Furthermore, the signatory must be an acting notary.

The existence of a type 9 entry marks the termination of a decision thread and identifies the final entry. The termination occurs when the processing time runs out and only then a type 9 entry may be produced and is formally correct. There should be only one (properly notarized) type 9 entry to a terminated thread.

Furthermore, there is an order in which notaries attempt to create and notarize a type 9 entry for a terminated thread. Firstly, the list of notaries involved in this process consists of all notaries who are acting at the moment the processing time for the thread runs out (termination time). The list begins with those who also participated in the notarization of the last entry of the thread, following the order the signatures were placed in the notarization. The remaining notaries are added in accordance with their notary numbers. Now the first notary in the list creates and signs a type 9 entry starting a notarization process. If a notarization does not occur within the maximal notarization time (for this lineage) after termination time for the particular thread, then the attempt has failed by definition and the next notary in the list must attempt to create and notarize a type 9 entry and so on. Each notary has a time window which is in size equal to the maximal notarization time (for this sub-lineage) and these windows are disjoint.

The above procedure applies only if a type 9 entry is produced before the respective sub-lineage is "cut off" (according to the definition of the type 1 entry). Otherwise, a type 9 entry must be produced by the first notary of a follow-up sub-lineage and notarized following normal time constraints and procedures.

C.13 Type 10 - Liquidity transfer note

A type 10 entry has the following structure:

1. The first byte is 0x29.
2. The following 20 bytes are the complete ID of a notarized entry of type 5 or type 15. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type (1)5, which contains the description of the respective currency/obligation.

3. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity to be transferred (transfer amount). This value must be non-negative and, if strictly positive, cannot be below the minimal transfer amount for the respective currency.

4. The following 20 bytes are the complete ID of a notarized entry of type 14. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 14, which must be free and carry a sufficient liquidity amount (of the same currency/obligation) to cover this transfer. The holding participant in this type 14 entry is also the future signatory of this entry, i.e. is the one who sends the liquidity.

   Alternatively, one may specify here the complete ID of a notarized entry of type 11. This is possible if the currency/obligation specified above is an obligation (type 15) and the signatory of that obligation is the same participant as the one defined by this type 11 entry, who shall also be the future signatory of this type 10 entry. Also, in this case the transfer amount must be positive. If the transfer amount is zero and a type 11 entry is supplied here, then this type 10 entry amounts to a transfer request to the account specified by the type 11 entry and the currency/obligation is arbitrary.

5. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the recipient of the liquidity.

   If there is no recipient, i.e. the liquidity is to be deleted from the system, all 20 bytes are set to 0x00. In this case, but also in the case a type 11 entry is specified as above (under 5.), the transfer amount must be positive.

   Alternatively, instead of specifying a recipient one may specify a type 5 or a type 15 entry (by providing the complete ID of the respective notarized entry). This means that the liquidity specified under 2. and 3. above goes to whoever provides a certain amount of liquidity (specified below) in this other currency. The two currencies/obligations are different if and only if the transfer amount under 3. is strictly positive.
   Until someone satisfies the request the signatory of this entry is considered to be the only eligible recipient under this type 10 entry. Otherwise the only eligible recipient becomes the signatory of a future type 10 entry which refers to this type 10 entry under 5. Note that at that time this type 10 entry might effectively carry a lower amount than the transfer amount defined above (due to a discount).

   As already indicated above, instead of specifying a recipient or a liquidity/obligation one may, as a third option, specify another type 10 entry (by providing the complete ID of the respective notarized entry). This is only possible if that other type 10 entry specifies a currency/obligation entry under 5., which is equal to the currency/obligation of this entry, and the transfer amount under the current type 10 entry is equal to the requested amount in the other type 10 entry. If everything is consistent the recipient under the current type 10 entry is defined as the signatory of that other type 10 entry.

6. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount requested in another currency/obligation, in case such currency/obligation was specified under 5. above. If a type 11 entry or a sequence of zeros was specified under 5. then this number must be set to 0. In all other cases this value must be positive. Furthermore, if another type 10 entry was specified under 5. then this number must be equal to the transfer amount of that other type 10 entry.

7. The last byte is 0x04.

A type 10 entry must be signed by the signatory defined in the entry and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the resulting type 12 entry to be formally correct it must be signed with the same key as the one identified as the signatory key in the underlying type 10 entry.

Note further that not all entries which have been notarized can be used by the notaries to collect fees. Only type 10 entries are used to this end. The notarization fee is applied to the transfer amount, such that, from the point of view of the recipient, this type 10 entry effectively carries a lower liquidity amount than the nominal transfer amount mentioned in the entry itself due to withheld fees (in particular).

Note that the notarization fees initially assigned to a notary (the first notary who signs a type 10 entry, which is then successfully notarized) are distributed among all notaries which were acting notaries during his or her tenure. This distribution depends on the decision threads which took place during his or her tenure and can be reconstructed by every observer. In particular, notaries must calculate and update these distributions in order to automatically assess the validity of claims due to notarization fees.

To other entry notarizations no fees are applied, which does not mean, however, that they can be treated any differently by the notaries. Notaries can "discriminate against" entries, which are subject to notarization, only if these entries appear to be unimportant or unnecessary or if the same legitimate objective can be achieved with much simpler or fewer entries or if there are systemically more important entries which have to be considered first. In particular, notaries can refuse to process entries which appear to be spam or part of an DoS - type attack. There are various justified measures notaries can (collectively) take to avoid processing or having to store to many entries while keeping possible inconveniences to the community to a minimum:

- Refuse to process type 10 entries with very low amounts.
- Process only a maximal amount of operation proposals per operator per unit of time.
- Refuse to process entries in a decision thread which are submitted almost immediately after the previous entry in the respective decision thread.
- Refuse to process type 10 entries which have extremely recent predecessor entries.

Apart from having the above structure a type 10 entry must be pointing to formally correct and fresh type 13 notarized entries to be formally correct as well. Also, the signatory must have a valid claim to the liquidity as described above. The entry is not formally correct if the recipient under this entry is specified via another type 10 entry the signatory of which already reclaimed its liquidity using a type 14 entry.

C.14 Type 14 - Liquidity claim

A type 14 entry has the following structure:

1. The first byte is 0x2D.
2. The following 20 bytes are the complete ID of a notarized entry of type 5 or type 15. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type (1)5, which contains the description of the respective currency/obligation.
3. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the holder of this liquidity (holding participant).

4. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity that is held by the owner specified above (total liquidity amount). This value must be non-negative.

5. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the amount of liquidity that is the part of the above liquidity that the owner can only use as referee for stakes in decision threads (non-transferable amount). This value must be non-negative and cannot exceed the total liquidity amount.

6. The following 2 bytes are an unsigned short integer denoting the number of predecessor entries which the liquidity for this entry comes from, or, more precisely, which in total define the liquidity currently held by this participant. Let n be this number.

7. The following $8 \cdot n$ bytes are IEEE 754-1985 binary64 floating-point numbers denoting the liquidity impact coming from each of the n entries. These values can be positive (e.g. in case liquidity is received from a source) and negative (e.g. in case liquidity was sent to someone). The sum of these values must be equal to the total liquidity amount.

8. The following $20 \cdot n$ bytes are the complete IDs of the aforementioned n notarized predecessor entries. Technically, these entries are of type 13, but are wrapping entries of other types.

9. The following n bytes are numbers between 0 and 255, denoting the scenarios according to which the liquidity impact coming from the predecessor entries is occurring. There are in total 10 scenarios specified below. Accordingly, only numbers between 0 and 9 are possible here.

10. The last byte is 0x04.

There is a limited number of scenarios which imply liquidity claims:

- Scenario 0: The respective predecessor entry in this base must by of type 14 and the associated liquidity amount must be non-negative. Furthermore, there can be at most one scenario 0 predecessor entry in a type 14 entry and there should be one in case there are already type 14 entries for this participant and currency/obligation. This means that properly notarized formally correct type 14 entries for a given holding participant and currency/obligation form a sequence.

- Scenario 1: In this case the respective predecessor entry must be a type 10 entry and the recipient under this type 10 must be the holding participant specified above. Furthermore, the associated liquidity amount must be positive.

- Scenario 2: In this case the respective predecessor entry must be a type 10 entry and the signatory under this type 10 must be the holding participant specified above. Furthermore, the associated liquidity amount must be negative.

- Scenario 3: The holding participant is a notary who was the first to sign in an amount of notarized type 10 entries and is claiming his or her fees for this service or the signatory is a notary who possibly was not the first to sign the respective entries, but is claiming a part of the fee initially reserved for the first signing notary, because of one or more won decision threads. Under this scenario the predecessor entry is a type 13 notarized entry of a type 11 entry, which contains the public key of the first signing notary. The associated liquidity amount must be positive. Also, the amount must be such that all type 10 entries with the same first signing notary are considered.

- Scenario 4: The holding participant is a referee who is claiming his or her initial liquidity which is granted to every newly appointed referee. In this case the predecessor entry is wrapping an entry of type 9 (termination entry of the respective decision thread which appoints this participant as referee) or an entry of type 5 (currency description entry which must define this participant as the first referee). The associated liquidity amount must be positive.

- Scenario 5: The holding participant is an operator who is forfeiting a certain liquidity amount attached to an operation proposal under parts 6. and 7. of the respective entry of type 7. However, the operator can reclaim this amount if the proposal is granted. This functionality is optional and can be used by the operator to indicate the seriousness of his or her proposal to the referees (and/or the notaries). Under this scenario the predecessor entry is of type 7 and the associated liquidity amount is negative.

- Scenario 6: The holding participant is a referee who is attaching a stake to a decision thread to take part in it. In this case the predecessor entry is of type 8 and the associated liquidity amount is negative.

- Scenario 7: The holding participant is a referee who is claiming a fee for the approval of an operation proposal or a stake from another referee due to a won decision thread. In this case the predecessor entry is wrapping an entry of type 9 (termination entry of the respective decision thread). The associated liquidity amount must be positive. Note that the claim might come from a participating referee or a referee who is the appointing referee for one or more of the participating referees etc. The associated liquidity must include all liquidity claimable according to this scenario for this type 9 entry and holding participant.

- Scenario 8: The holding participant is a referee who is claiming his or her own stake back due to a won decision thread. In this case the predecessor entry is wrapping an entry of type 9 (termination entry of the respective decision thread). The associated liquidity amount must be positive.

- Scenario 9: The holding participant is an operator whose operation proposal was approved. In this case the predecessor entry is wrapping an entry of type 9 (termination entry of the respective decision thread). The associated liquidity amount must be positive. Note that the operator claims not only the amount requested in the proposal but also the liquidity initially attached to the proposal (which was temporarily lost).

Note that for scenario 3 above the fees must be claimed within one notary tenure after the respective first signing notary's tenure has ended. For scenarios 4, 7, 8, 9 a claim must occur within one notary tenure after notarization of the respective predecessor entry. There are no time limits for the other scenarios. When claiming (positive) liquidity according to some scenario the maximal claimable amount at the respective moment must be claimed for the resulting entry to be formally correct.

In most scenarios the associated liquidity amount must be equal to the actual liquidity impact of the predecessor entry at the time the predecessor got notarized, but discounted with the deletion/growth rate specified in the description of the respective currency/obligation. Remark that scenario 1 is a special case as notarization fees are applied, which shrinks the effectively carried amount (to which in turn the deletion (or growth) rate is applied). Furthermore, for scenario 3 the discount is not applied to the whole time period between the notarization of a type 10 entry and the creation of this type 14 entry, but only from the moment on the tenure of the respective first signing notary has ended. Also, note that liquidity that is "tied up" in a decision thread does not shrink, which means in particular that for a claim according to scenario 8 the claimable amount is equal to the deposited stake (at least at the time the decision thread terminates) rather than its

discounted version. In other words, the discount is applied only once the thread has terminated and liquidity becomes claimable. The same applies to scenarios 7 and 9.

Note further that the liquidity amount currently carried by a type 14 entry is not necessarily the total liquidity amount specified above, but its discounted version depending on how much time has elapsed after its notarization. The actually carried amount can be smaller or larger depending on the rate.

Note that while the liquidity amount coming from each predecessor entry is mentioned in the type 14 entry, the impact on the non-transferable amount is not included. However, this information can be extracted from the predecessor entries (or their predecessor entries etc.) such that one can verify that the non-transferable amount claimed in the new type 14 entry is consistent with the past: There is no non-transferable liquidity involved in scenarios 1, 2, 3, 5, 7 and 9. In other words, only the scenarios 0, 4, 6 and 8 involve non-transferable liquidity. For scenario 0 the non-transferable liquidity is simply stated in the preceding type 14 entry (it must be discounted, however). For scenario 4 all liquidity is non-transferable. For scenario 6, the amount of non-transferable liquidity (that is lost) depends on how much non-transferable liquidity was carried by the type 14 entry referenced in the type 8 entry. For instance, if it carried more than the stake, then only non-transferable liquidity was used for the stake, otherwise only a part of it (which is equal to all remaining non-transferable liquidity carried by the respective type 14 entry) was non-transferable. Finally, in case we have scenario 8 the non-transferable liquidity that is reclaimed depends on how much non-transferable liquidity was in the stake for the respective type 8 entry through which this referee took part in the thread.

A type 14 entry must be signed by the holding participant defined in the entry and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the resulting type 12 entry to be formally correct it must be signed with the same key as the one identified in the entry. Apart from having the above structure a type 14 entry must have formally correct type 13 notarized predecessor entries to be formally correct as well. Except for scenarios 2, 5 and 6, these entries must be even freshly notarized entries. Finally, the type 14 entry must involve all predecessor entries which have diminished liquidity owned by this participant (scenarios 2, 5 and 6), unless such entries have already been included in preceding type 14 entries. While including such entries their full impact must be taken into account, i.e. the associated liquidity amount must be equal to the amount according to which the total liquidity owned by this participant was diminished (whereas a discount in time might have to be applied). For the scenarios with a positive impact the impact should be as large as possible, i.e. full liquidity should be claimed.

We call a properly notarized type 14 entry *free*, if there are no type 7, 8, 10 or 14 entries pointing to it.

C.15 Type 15 - Obligation description

A type 15 entry has the following structure:

1. The first byte is 0x2E.

2. The following 20 bytes are the complete ID of a notarized entry of type 11. Technically, it is pointing to an entry of type 13, which is wrapping an entry of type 12, which is wrapping an entry of type 11, which contains the public key of the signatory of this obligation. The signatory has the right to issue new liquidity for this obligation.

3. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the basic rate at which all outstanding liquidity grows or is deleted (annual growth or discount rate, e.g. 5 standing for 5% growth or -5 standing for 5% shrinkage).

4. The following 8 bytes are an IEEE 754-1985 binary64 floating-point number denoting the minimal amount of liquidity to be transferred in standard liquidity transfers.

5. The following 8 bytes are an unsigned long integer defining the length of part 5. (in bytes) below:

6. The following part is the actual description of the obligation and contains the definition of who is obliged to do what if some amount of liquidity of this obligation is deleted by a participant (depending on the amount) and defines the standard operating procedures for the settlement. The default interpretation of this byte sequence is the UTF-8 code, but an interpretation as a binary file is also possible if it is clear from the context how the byte sequence is to be interpreted.

7. The last byte is 0x04.

A type 15 entry must be signed by the signatory and the resulting type 12 entry is sent to the notaries for notarization and dissemination. For the type 12 entry to be formally correct, it must be signed with the same key as the one mentioned under 2. in the underlying type 15 entry.

Type 5, type 15 and type 11 entries are different from all other entries in that they do not have to be renotarized. I.e. once notarized they remain fresh for ever by definition.