Peer-to-Peer Securities Trading

Dr. Nick Gehrke, Andre Daldrup, Lutz Seidenfaden
Institute for business informatics, Dep. II
University of Goettingen, Germany
{ngehrke | adaldru | lseiden}@uni-goettingen.de

Abstract

Peer-to-peer systems are highly distributed and self-organizing communication networks. They get on without a central server as much as possible. This paper proposes to organize the trading of securities - like it is being done up until now at the stock exchange - on a peer-to-peer basis. In this concept there is no central server that takes over the matching of supply and demand or rather no central server that administers the order book. The result is a highly decentralized peer-to-peer network. Ideally, every trader will be represented through his peer, which carries out the buying and selling contracts in coordination with other peers. The network is more robust in case of a system crash than a centralized system because it works without a central coordination unit. Furthermore costs for central system components can be partly dropped. In addition it is possible that everybody can participate in the trading of securities and does not rely on a bank. The Chord protocol is being used as a decentralized coordination mechanism.

1. Introduction

This paper presents a peer-to-peer infrastructure with which the trading of securities can be handled mostly in a decentralized way. The matching of orders and the trading of securities can be handled in a decentralized way. We will show that, at this stage, there are still certain central elements that one can't do without. The motivation to use a peer-to-peer market model is that it is more robust to the break down of system components than centralized systems and the fact that costs for a central stock exchange platform can partly be dropped, e.g. maintenance costs for central components are distributed among the participating peers. From a user perspective the proposed system has the advantage that one can trade securities without paying transaction fees to the intermediary (e. g. banks).

This paper is organized in the following way: First of all, we will have a closer look at related work in this thematic area. The following basic part will briefly show how today's computer stock markets work without peer-to-peer solutions. This will be shown using the example of the German Xetra® system. The peer-to-peer market model which will be presented in the paper is based on Chord. Therefore the Chord protocol will be briefly explained.

The following sections concern themselves with the concrete concept of a peer-to-peer stock exchange. At first, requirements will be defined and then, assumptions of the peer-to-peer stock exchange will be formulated. The rest of this contribution will describe the gradual conduction of the transaction of securities. Finally, the presented model will be criticised and it will be shown what researches might be done in this thematic area in the future.

2. Related Work

Applications and market models which are based on peer-to-peer technology are at the centre of today's research, whereas most of the researches have been interested in the construction of electronic marketplaces [5] and file-sharing systems [7,12]. Researches that concern themselves with the area of financial-economic applications and especially with the area of trading in securities supported by peer-to-peer technology without the central instance of a stock exchange, have not yet been discussed extensively.

Lohmann et. al. did develop an electronic trading system which is organized in a decentralized way, but they built up the prototype using software agents as a basis rather than using peer-to-peer technology [10].

Schmidt and Vinowski show one possibility how to disconnect the relation between client and server using the example of an electronic matching system, whereas they put the strain mostly on

technical details and economic facts are not taken into consideration extensively [14]. Maxemchuk and Shur describe a decentralized stock exchange trading system that is based on a timed reliable multicast protocol (TRMP) [11]. In practice, a company called Liquidnet created a decentralized trading system based on peer-to-peer technology. However, this system is specialized in the institutional trading of big orders and it predominantly supports the buy side [6,8].

3. Principles of Trading and Chord Protocol

3.1 State of the Art Financial Securities Trading

The process of a security transaction can generally be divided into the following four phases [1]:

- Information and advice
- Registration of the order and its routing
- Matching and price determination
- Clearing and settlement

A fully electronic trading system should support the phases (2) to (4). The first phase of information and advice will only be mentioned and explained here because of reasons of completeness. Afterwards, the following three phases will be described using the example of the continuous trading of the trading system called Xetra® (Exchange Electronic Trading) used by the Deutsche Börse AG.

Information and Advice

The first phase consists of the investors' decision in what to invest, and then its estimation of the chances on the market. In order to do this, the investor needs information which is provided in form of quotations that can be obtained directly from the stock exchange or from other financial service providers (i.e. Reuters). On the basis of the market value information the investor got this way or through other business data, a positive or a negative decision to buy or sell can be established for a certain security. Nonetheless, in the following conception of this electronic trading system, this phase will not be taken into consideration anymore.

Collection of Data and Its Routing

The recordation of order and its routing is established through so-called Xetra® Clients. They transfer the order to the electronic order book. Xetra® is based on Client-Server architecture and offers multiple forms of binding (see figure 1). The Xetra® front-end for trading members consists either of a single Xetra® Client or of many clients which are connected through the Member Integrated System Server (MISS). Additionally a binding of inhouse systems of members to Xetra® is also possible through the interface Values API (Virtual Access Link Using Exchange Services - Application Programming Interface). The backend of Xetra® is represented through a server for the stock exchange system where the central stock exchange functions are carried out. [1,4]

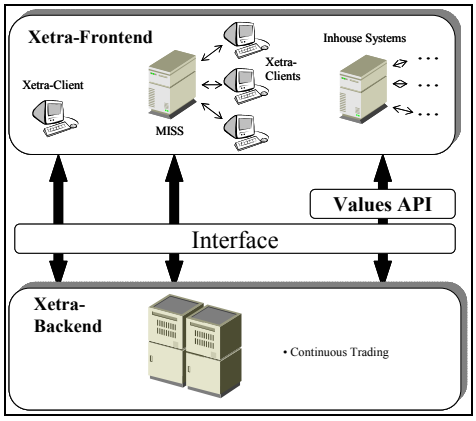

Figure 1: Xetra®- Architecture [1]

Matching and Price Determination

This phase is the core of the trading system where the bids and asks are confronted in the order book, to automatically conduct the matching of the opponents and the price determination. The bid price is the amount of money a buyer would spent to purchase a security and accordingly the ask price is the quotation a seller would sell his securities for. For reasons of simplicity, the following text takes only the basic order types that can be handled with Xetra® in the continuous trading ("Market Order", "Limit Order" and "Market-to-Limit Order") into consideration. An abstraction will be made concerning further specifications of orders through additional conditions of execution, validity or trading limitations.

- *Market Orders* are unlimited buying and selling instructions that will be carried out at the next price determination.
- *Limit Orders* will be carried out at the certain limit or at a better price.
- *Market-to-limit Orders* are unlimited orders that will be carried out at the best limit in the order book, as far as the limit is represented through at least one limited order and no market order is present in the order book. Those parts of a market-to-limit order which are not carried out will remain in the order book as limited order, whereas the quotation of the conducted part is set as the limit. [3]

The Xetra® system differentiates the phases pre-trading, trading und post-trading. Figure 2 shows the flow of trading of Xetra® in the continuous trading.

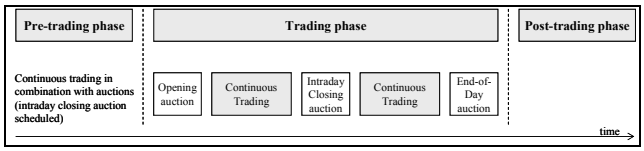

Figure 2: Flow of trading [3]

In the pre-trading phase, participants in the market are able to add orders or to change or delete existing orders. The order book is closed in this case and it shows either the latest fixed price or the best bid or ask price of the last trading day. The trading phase starts with an opening auction in which all open orders of the trading day before and all orders that have been entered in the pre-trading phase are included.

The quotation is determined through the principle of most execution that means the quotation, to which the most transaction volume can be established, is fixed according to the limits of the potential sellers and the demanders. The intraday closing auction which interrupts the continuous trading and the end-of-day auction which finishes the trading phase run analogous to the opening auction. In the post-trading phase the participants in the trade are able to enter new orders, or to either change or delete existing orders, just like in the pre-trading phase. In the following text only the continuous trading in the trading phase will be taken into consideration.

During the continuous trading the order book is open in order to show the limits with their belonging cumulating volume as well as the number of orders per limit. Every new order will be checked if a fitting opposite order in the order book exists. In doing so, the order is processed completely, in parts or not at all. The execution is done with a price/time priority so that orders to buy with a higher limit are preferred to orders with a lower limit. In the case of orders to sell the lower limits are given priority. The time is the second criterion when it comes to an identical limit which leads to a carrying out of orders following the first-in/first-out principle. Altogether, market orders are given priority over limited orders. After the business transaction both parties receive a carrying out receipt. [3]

During the determination of the price in the continuous trading it will be tried to carry out the incoming order immediately that means that according to the order limit, a counterparty is being searched for. Certain rules have to be taken into consideration when certain constellations appear.
These are the basic rules:

- If a market order gets into the order book in which only market orders exist, the reference price that corresponds to the quotation of the last transaction will be taken exclusively to set the order's price.

- If the order book consists only of limited orders either the highest buying or the lowest asking price determines the price if market orders, market-to-limit orders or limit orders come in.
- Market-to-limit orders are being refused in an order book in which no orders, only market orders or market and limit orders consist. The following figure shows an example for the matching of the counterparties and price determination. Here, the incoming limited selling order is carried out against the market-buying-order on the left hand side of the order book. [3]

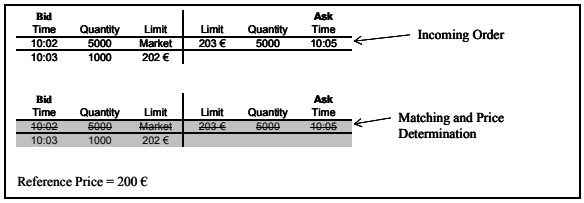

Figure 3: Matching and Price Determination [3]

Clearing and Settlement

Clearing and settlement is the transaction of securities which consists of the settlement, that is the finding out of the demands of both contracting parties and the fulfilment of the obligations money- and number-wise. In the case of the Xetra® System the carrying out information is transferred to the Cascade-System (Central Application for Settlement, Clearing and Depository Expansion) via Xontro Trade which is the interface between the phase of agreement and the carrying out phase (number and money-wise). Xontro Trade allows the recording and the forwarding of carrying out information and creates the settlement note. The Cascade-System guarantees a secure processing of transactions in the form of bit-by-bit fulfilment. This means that a buyer will only receive the securities after having paid for them [1].

The remainder of this paper presents the basic idea of P2P securities trading, whereas for reasons of simplicity only limited orders and not all of the basic order types mentioned above are taken into consideration.

3.2 Chord in a Nutshell

Before we take a closer look at the construction of a P2P-based stock exchange trading, there will be a brief description of the Chord protocol [2,15], because the mechanisms of Chord are the basis for the following text. The first thing is to explain the goals of Chord. To provide a better understanding, the following text explains how Chord achieves these goals.

Chord works without a central index server and without super nodes. In a peer-to-peer network which is based on Chord, every peer has a unique peerID with the bit length of m. The peerID has been processed pragmatically as the IP of the peer is transformed into a peerID through a hash algorithm. The resources of the peer-to-peer network do have a resourceID of the same bit length as the peerID. In the case where a resource corresponds to a file, the resourceID can be determined through the calculation of the file's hash value for example. Chord does a very elementary job. However, it is very efficient:

Chord finds the resourceID for the responsible peer with the peerID p in logarithmic runtime. This runtime corresponds to the number of peers which have to be used as intermediaries ("hops") to locate the responsible peer.

If the number of peers in the network increases, Chord is able to scale very good using logarithmic runtime. Chord has such good runtime abilities because the peers in the network refer to one another intelligently which makes the routing highly efficient. In principle, every peer in the network contains a reference to the following peer. The following peer is the peer which has the next higher peerID. The peer with the last peerID is connected to the first peer, which establishes a so-called Chord ring (see figure 4a).

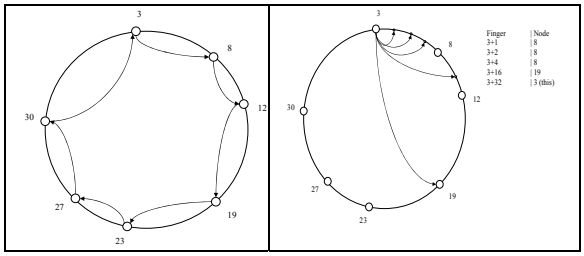

Figure 4a: A Chord ring with successor relations, 4b: A peer with a finger table

The figure in 4a depicts the general structure of a Chord ring. Every peer knows the successor peer in the space of the peerID and every peer is responsible for a resourceID if resourceID=peerID or resourceID>peerID and the peerID of the successor peer is bigger than the resourceID. But in this case the routing is inefficient: If the peer with the peerID 3 wants to find the resource with the resourceID 31 (which is based on the peer with peerID 30) it has to work gradually through all neighbouring relations. The expense to search is proportional to the number of peers on the net. In order to make the search more efficient, every peer has a routing table with a constant number of entries (or a finger table). This means that the finger i points to the peer which is responsible for the resource ($peerID+2^i$) (see 4b). If the peer searches for a certain resource, it passes on the inquiry to the next peer which is in front of the resource nearest in the ID space. Then this peer passes on its inquiry using its finger table until the responsible peer has been hit directly. The construction of finger tables reinforces the connection between the peer and the peerIDs, which are located close to the ID space of the peer itself. The distance of the finger in the ID space is heavily increased by an increasing finger order i. If a resourceID is located far away from its own peerID, then it's possible to make an inaccurate but big step through the ID space using a finger of a higher order. The inquiry is passed on to a peer nearby. However, this peer nearby is more likely to have a finger table which contains the peer one has searched for, or at least it contains an ID which is closer to the peer one has searched for. Every peer has a finger table which has been constructed in such way. This is the reason why the inquiry is always passed on to a peer which is "closer" in relation to the resourceID. This results in a fast search with logarithmic runtime.

For a more detailed discussion of constructional algorithms of the Chord ring, as well as of finger tables and of algorithms of stabilisation, see [15]. For further information about routing protocols or about overlay structures, compare [2, 9, 13, 16].

4. Securities Trading Using P2P

4.1 Requirements for Peer-to-Peer Financial Securities Trading

A peer-to-peer based trading system should analogously fulfil certain technical and functional requirements to ensure the acceptance of all participants in trading and therefore to make a practical usage of the system possible. The requirements that have to be taken into consideration are mainly the following [10]:

- Performance
- Decentralization /Availability
- Scalability
- Safety
- Extendibility.

The matching of the counterparties is the core of the trading system so that the performance is determined through the period in which matching transaction partners are found and adjusted against each other.

Trading systems have to have a permanent availability in the course of trading. Therefore it has to be made certain that no central components are in the system, but that a wanted redundancy is achieved. In this case, if the system crashes it can be switched to another component that has the same range of functionality. Additionally, the number of users in the trading system must not be limited or

fixed but it should ideally be able to admit an unlimited number of users. That means the system has to be scalable. An important criterion for the acceptance of the system is its safety. It has to be ensured that there is no possibility of manipulation or that no fake orders can get into the system. Every user has to be identifiable at the end of the business transaction. Due to economic reasons a possible extension of the system has to be considered (safety of investment) so that further functions (other kinds of trade or other nuances of trade) can be integrated. Furthermore, the possibility should exist to pair with other (trading) systems.

4.2 Assumptions

In order to guarantee a safe trading of securities these assumptions are assumed to be known:

- Every peer possesses an asymmetric pair of keys. A Trustcenter safes the identity "behind" a Public Key.
- Every peer possesses a business card or a certificate that has been signed by a Trustcenter. The business card contains the identity of the user and the Public Key of the peer.
- The transactions in the peer-to-peer network are continuous trading transactions.
- Neither "best" nor "cheapest" orders (market orders) are being dealt with but only orders with a concrete price (limit order).

4.3 Phases of a Peer-to-Peer Securities Transaction

Before a transaction in a peer-to-peer securities transaction is described in more detail, the phases that take place in every transaction will be shown. Figure 5 shows the phases that take place:

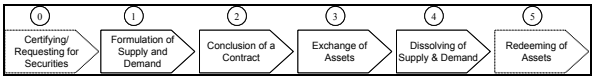

Figure 5: Phases of a peer-to-peer securities transaction.

In particular these are the steps that take place during a transaction. The phases 0 and 5 do not belong to the transaction directly:

0. *Certifying/Requesting for securities*. Before securities in the peer-to-peer network can be traded; the traders have to get their stocks and their amounts of money "certified" by their banks. A packet of certified securities, for example, is an issued file that will be exchanged into an amount of money with the counterparty after the conclusion of a contract. It is important that this does **not** have to happen before every single transaction. Packets of stocks and money can be certified at any time before the actual transaction takes place. Therefore, no contact with the bank is necessary before a transaction. This means that this phase is not directly belonging to the transaction.
1. *Formulation of supply and demand.* At the beginning of a transaction a buyer (of securities) or a seller (of securities) respectively formulates his or her bid [consisting of International Securities Identification Number (ISIN), price and amount]. Because there is no central unit, the bid will be stored on another peer in a decentralized way (for further details, read the next sections).
2. *Conclusion of a contract.* If the buying and selling bids fit to each other the contract is concluded. This contract will be stored on a peer in a decentralized way as well.
3. *Exchange of assets.* After the contract has been concluded, the money is exchanged for the goods and vice versa among the peers.
4. *Dissolving of supply and demand.* After the goods have been exchanged, the original bids will be removed out of the peer-to-peer network in order to guarantee that multiple contracts cannot relate to the same bid (so that one packet of securities would be sold several times).
5. *Redeeming of assets.* After the goods have been exchanged the parties of the contract own an amount of stocks or an amount of money that has been certified. After the contract has been presented to the bank, the amounts of stocks and money will be booked to accounts or to depots respectively. Redeeming must **not** take place directly after the transaction but may hap-

pen at any time after the transaction, which means that this phase does not belong to the transaction as well.

4.4 Actors

Before a technical description of a peer-to-peer network will be given, the persons concerned will be described. In this text it will be distinguished between permanent and temporary persons concerned. The permanent participants are the direct actors of the peer-to-peer transaction:

Permanent Actors ("Online-Actors"):

- **Peers**. The peers are the permanent actors during the securities transactions. A differentiation between the person who demands and the supplier of securities will not take place. Every trader is represented through a peer.

Temporary Actors ("Offline-Actors", "Back Office-Actors"):

- **Trustcenter**. The Trustcenter is only necessary to hand over the asymmetric pair of keys and the business card to the persons concerned or rather to the peers. Because the business card has been signed digitally by the Trustcenter, its authenticity can be understood by every participant without having to contact the Trustcenter.
- **Securities Bank**. The Securities Bank takes care of the deposits of all the participants. If required a Securities Bank can hand out digitally signed files to the traders in securities. These files are called "certified securities". With the digital signature, the authenticity of the certified securities is guaranteed. A Securities Bank is able to book the certified securities back to a deposit after a transaction. This may happen asynchronous, which means it does not have to happen directly after a transaction.
- **Money Bank**. A Money Bank has the same purpose as the Securities Bank but amounts of money and not securities are being certified at the Money Bank. This is relevant for persons who have a demand for securities. A bank can be a Securities Bank and a Money Bank at the same time. Several Money and Securities Banks can exist.
- **Adjudicator**. The Adjudicator will be called on if one of the persons involved in the transaction thinks that the securities transaction has not taken place orderly. The purpose of the Adjudicator is to discover the lapse of participants in the market. The function of the Adjudicator can be integrated as a function in every bank which means he is redundantly present.

4.5 Conducting the Transaction

The following text describes how a securities transaction in a peer-to-peer network takes place. The phases 0-5 from the figure 5 above will be described in technical detail.

Phase 0: Getting Money and Securities for Trading

In phase 0 the participants in the market get their certified securities and amounts of money from their Securities/Money Bank. Any number of volumes can be called. The bank digitally signs the certified securities/amounts of money in order to avoid peers creating their own securities or amounts of money. During the transaction the certified securities/amounts of money are being exchanged (see below). The figure number 6 shows a packet of securities that has been digitally signed and certified by a Securities Bank. A peer saves the data that has been handed out by a Securities/Money Bank locally. The procurement of certified securities/amounts of money does not have to happen directly before a transaction. This means that stocking up is possible and useful.

```
<security>
    <data>
            <ISIN>34758345</ISIN>
            <quantity>50</quantity>
            <issued>15.03.2003</issued>
            <validuntil>20.03.2003</validuntil>
            <issueID>66745674356723<issueID>
    </data>
    <seal>
     <issuer>internationalassetbank</issuer>
          <hash>45r45z6z4h665h65h56hhg8</hash>
           <signature>7ztz8945z6456z3465</signa
      ture>
    </seal>
<security>
```

Figure 6: A file that certifies a packet of securities; certified amounts of money look equivalent.

Phase 1: Proclaiming Demand and Supply

The first step of a peer-to-peer transaction is that a trader declares demand or supply of securities (orders). Because there is no central unit in which supply and demand come together, the order has to be saved decentralized on a peer. A simple suggestion would be to calculate the hash value of the International Securities Identification Number (ISIN). Then the saving of the order could be at the peer that is responsible for the calculated hash value. The finding of the peer is guaranteed through the Chord protocol, but the danger exists that only one peer is always responsible for the same ISIN which would lead to a bad load balancing within the peer-to-peer network.

To ensure a better sharing of the orders, the following steps are to be done:

1. The ordering peer calculates the hash value of an order. This is done by calculating: `Hashvalue („ISIN"+"PRICE"+"AMOUNT"+{„0", if demand | "1", if supply })`

Adjacently the ordering peer calculates a symmetric key using the information of its order as parameter. The method to calculate the key can be a hash function as well (although it has to differ from the hash function in (1)).

`2. CalculateKey(„ISIN"+"PRICE"+"AMOUNT"+{„0", if demand | "1", if supply })`

The ordering peer encodes its order with the help of the calculated symmetric key from (2). The ordering peer saves its order on the peer that is responsible for the hash value from step (1). Figure 7 exemplary shows a demand order.

```
<demand>
      <data>
       <ISIN>65746756</ISIN>
            <quantity>70</quantity>
            <price>456.00</price>
            <issued>17.03.2004</issued>
            <validuntil>20.03.2004</validuntil>
            <tan>43534534543</tan>
      <temppubkey>tz5776356634765<temppubkey>
      </data>
      <hashvalue>t56t34545t4356</hashvalue>
      <signature>u85z87658z5468</signature>
</demand>
```

Figure 7: Example of a demand order; a supply order looks equivalent.

An order contains the ISIN, amount, price and different date tags. The TAN of the order is calculated the following way:

```
TAN=Hashvalue("PUBLICKEY"+"ISIN"+"QUANTITY"+"PRICE"+"ISSUED"+"VALID- UN-
TIL")
```

The purpose of the TAN is to identify the order clearly. `publickey` is the Public Key of the ordering peer. An analogous storing of the order to the one described above takes place. The order will be encoded with the Symmetric Key `CalculateKey („TAN")`. Adjacently the encoded order is stored on the peer that is responsible for the hashvalue `Hashvalue ("TAN")`.

In an order, there is a temporary Public Key stored. In this case, temporary means that it is only used for one single transaction (so it is not the Public Key of a peer that has been signed by a Trustcenter). As we will see later on, this Public Key is relevant for the settlement of a transaction. In the result, every order is stored twice. An order is saved using its coining (ISIN, amount, price). This saving lets (i.e.) someone who has a demand for securities find orders that have been saved by a supplier of securities. So this is important in order to start a transaction or rather to find adequate trading partners (ex ante). The storage using a TAN helps to find a unique order which is important if a transaction has been finished and one wants to find one unique order (ex post). Furthermore, the procedure that has been described above has the following characteristics:

Both kinds of storing save the encoded order to a responsible peer. That means that a peer on which the order is being saved will not be able to read the order due to its being encoded. Another peer that searches for the order knows the parameter of the search (like ISIN, amount and price) and is therefore able to find the responsible peer by calculating its hash value as well as creating the key to encode the order (the same way as an order is searched for using the TAN). The possibility that an order is manipulated is ruled out. If a peer manipulates the encoded order, the hash value noted in the order does no longer fit.

Furthermore, every order has been digitally signed by the creator of the order so that it can be found out which peer has created which order and which peer did not create it, when in doubt.

Phase 2: Sealing a Contract

After a peer has stored its order (as it is described in phase 1) on the peer-to-peer network, fitting orders of supply and demand have to be connected to each other. The result of fitting orders of supply and demand is called a contract. A contract is being sealed either by the supplying peer or by the demanding peer. In one contract, several orders can be connected. During the process of connection of different orders, the creator of the contract has got the possibility to block the certain decentralized order to prevent that one order is used in several contracts. Figure (8) shows a model contract that has been concluded by a supplier of securities.

```
<contract accomplisher="supplier">
  <data>

    <supplytan>67546746837678</supplytan>

    <demandtan>5463563475345</demandtan>
    <demandtan>43653476543856</demandtan>
    <temppubkey>z5z78476345t56z5</temppubkey>

          <issued>15.03.2004</issued>
  </data>
      <hash>th854z754z5t454</hash>
      <signature>56784z76767676</signature>
</contract>
```

Figure 8: A contract that has been concluded by a supplier of securities.

All TANs of the order that are an item of the contract are shown. It is obvious that every single order has to fit to each other according to its ISIN, amount and price. This is something the creator of the contract has to take care of. Additionally his job is to search for the fitting orders. Because the supplier of securities is the creator of the contract, there is only one TAN that relates to a supply of an order.

The other TANs are orders of a demand whose sum creates the opposite part of the supply order. The contract has been digitally signed by the creator and a note is made of the hash value of the contract. The contract has to be saved redundantly on several peers. In order to let every contracting party know of the existence of the contract, it will be saved at those peers who are responsible for the hash value of the TANs or responsible for the orders of demand (i.e. `Hash-value („a"+„DEMANDTAN")`).

Accordingly, the key is derived out of the hash value of the demanding order's TAN. Furthermore, the contract contains a temporary public key from the creator of the contract that is necessary for exchanging the securities (see next section). Because the orders are stored on other peers in a decentralized way, it is not necessary that all contracting parties are online at the time of the conclusion of the contract. To be even more certain that the contract will be concluded correctly, it can be planned additionally that every contracting party has to sign the contract digitally. But this would demand the contracting parties to be online at the time of the conclusion of the contract and would therefore be a limitation. Furthermore it can be said that it is not possible to trace the trader of the order back from only having the TAN so that there is a certain anonymity between the contracting parties.

Phase 3: Exchanging Money for Securities

Finally, after the contract has been concluded, the securities have to be exchanged for the money. In order to do this, the certified securities or rather the amounts of money mentioned above are used (see figure (6)). First of all, the file gets encoded asymmetrically. For this, the temporary Public Key from the order or rather from the contract is used. Doing this ascertains that only those parties of the contract named in the contract are able to encode the certified securities/amounts of money. Then, the certified securities/amounts of money are stored on the peer that is responsible for the hash value `Hash-value („b"+„demandtan")` bzw. `Hash-value(„b"+"supplytan").` This means that every peer finds the exchanged goods at the corresponding peer. If the certified securities/amounts of money are not the ones of the appropriate order, the Adjudicator can be involved in order to clarify the transaction. Because of the digital signatures in every file, the Adjudicator is able to localise lapse and to call the responsible peer to account (i.e. exclusion from trade).

Phase 4: Deleting Demand and Supply Announcements

After the contract has been concluded the orders that have been used in the transaction have to be deleted or rather they have to be marked as no longer available, because no contract is allowed to exist, in which an order is object of a contract again. Deletion of orders can be done by the creator of the contract but additionally by at least one member of the opposite party which presents the (digitally signed) contract to the peers where the orders involved are stored. This can be seen as proof that the corresponding orders are really object of a contract. The peer that is responsible for the order has to delete the corresponding order. If it does not do that (out of spite), it might well be that this order becomes the object of a contract again.

This becomes obvious when the parties redeem their goods at the Securities/Money Bank (see next section). Then it is the Adjudicator's task to clarify and sanction the lapse.

Phase 5: Putting Money and Securities on the Bank

The last phase 5 is to be regarded equivalent to phase 0. In this phase the participants in the trade redeem their exchanged certified securities/amounts of money of a transaction at a Securities/Money Bank. This takes place after the certified securities/amounts of money, the contract and the order involved are presented. Then the Securities/Money Bank books the securities to a deposit or the amounts of money to accounts. The bank deals with the case that the certified amount of money is too high and "change" would become due. The certified securities/amounts of money will then be registered as invalid. If securities that are marked as invalid are presented for redeeming again, fraud would be obvious. Then it is the task of the Adjudicator to investigate the fraud.

4.6 Retrieving the Order book

After it has been described how a peer-to-peer transaction works, the text will explain briefly how a participant in the stock exchange is able to take a look into the order book. Because the orders are distributed and then saved among the peers in the peer-to-peer network, the order book or rather the information out of the order book has to be collected by the responsible peers. This can happen by

asking all the peers that are responsible for a certain combination of price and amount of a security iteratively. The peers that have been asked simply supply the amount of orders that have been stored on them. So the volume of a trade and the prices can be found out quite easily.

5. Criticism of the Model

The presented peer-to-peer trading network should only be seen as the first approach to handle transactions of securities in a decentralized way. There still remain weaknesses that need to be discussed. These might be some of the problems at hand:

- *Orderdistribution*. The peer that is responsible for the order is determined by four dimensions (supplying order|demanding order, ISIN, amount and price). If there are too many orders with exactly the same coining in all four dimensions the danger of an overload of a peer is present. If an overload is in sight, more mechanisms need to be developed that make a better load balancing possible.
- *Crashing of peers.* If a peer's computer crashes or the peer does not log itself off the network properly, the danger is at hand that orders might get lost, which means that they're simply deleted from the distributed order book. It might be helpful in this case that not only one peer is responsible for the storage of orders or other files, but that it might be several peers' task to do that. One suggestion would be to store all information at the n successor peers according to the PeerID within the Chord protocol. Having such a redundancy, several peers that are all independently connected to each other and that are not compulsorily physically neighbouring peers, would have to crash in order to get the same result mentioned above.
- *Lying peers*. Frauding peers could return false information [e.g. concering the report of numbers of orders (to calculate the order book)]. The point here is to keep all the information redundant as mentioned above. If one of the peers that normally keep the information redundant is lying, the discrepancy between the results of the other peers and its own result will automatically come to attention. An attacker would have to control all peers that keep the information redundant, but this is hardly possible because a peer is not able to pick the successor peer itself, due to the Chord protocol. If the successor structure within the Chord protocol is determined by the hash value of the IP of a peer, then it would be extremely unlikely that an attacker would be able to control several peers.

Finally, let's take a closer look at the requirements for a peer-to-peer network for the trade in securities. The requirements are fulfilled as follows:

- *Performance*. Performance is achieved mainly by the Chord protocol because it finds resources within the network using logarithmic runtime.
- *Decentralization/Availability*. Because it concerns a pure peer-to-peer network, a high decentralization is provided. A high availability is achieved due to a lack of centralization and a redundant storage of files on several peers.
- *Scalability*. Because every trader is represented through a peer, further traders are solely presented as other peers which means that the network can be extended in means of the number of participants.
- *Security*. Security is provided by cryptographic methods. It has been shown that information is stored decentralized on peers that are not involved in a transaction. In order to achieve that the peers will not be able to read the information, the data will be encoded.
- *Extendibility*. An extendibility of functions can be achieved easily. Theoretically, a trader would even be able to program his own peer. The cryptographic methods that are being used and other mechanisms to discover lapse (e.g. the Adjudicator) guarantee that a malicious peer cannot do any damage.

6. Conclusion and Future Research

In this paper a peer-to-peer network has been suggested in which trade in securities can be realised. Considerably parts of a transaction can be realised without a central unit. Future efforts should aim at making the process of a transaction even more decentralized. After all, a Securities or rather a

Money Bank is still needed to administer the securities. Another challenge will be the implementation of a prototype in order to test the possibility of the peer-to-peer market model that has been presented.

6. References

[1] Bodendorf, F. /Robra-Bissantz, S.: E-Finance, München, 2003.

[2] Dabek F., Brunskill, E., Kaashoek, F., Karger, D., Morris, R., Stoica, I., Balakrishnan, H.: Building Peer-to-Peer Systems With Chord, a Distributed Lookup Service, in: Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII), May 2001.

[3] Deutsche Börse Group: Xetra® – Market Model Stock Trading, http://deutsche-boerse.com/dbag/dispatch/de/binary/gdb_content_pool/imported_files/public_files/10_downloads/31_trading_member/10_Products_and_Functionalities/20_Stocks/50_Xetra_Market_Model/Xetra_Marktmodell_e.pdf, 2004.

[4] Deutsche Börse Group: Xetra® Trading Platform – Platforms and Configurations, http://deutsche-boerse.com/dbag/dispatch/en/kir/gdb_navigation/technology/20_Applications/10_Trading/10_Xetra_Trading_Platform?horizontal=page5, 2004.

[5] Gehrke, N. /Schumann, M.: Constructing Electronic Marketplaces using Peer-to-Peer Technology, in: Proceedings of the 36th Hawaii International Conference on System Sciences, 2003.

[6] Hegarty, R.: Leveraging the Napster Model: Peer-to-Peer (P2P) Computing Penetrates the Buy Side, in: Wall Street & Technology Online from Nov 27, 2000, http://www.wallstreetandtech.com/showArticle.jhtml?articleID=14704577, 2004.

[7] Leuf, B.: Peer to Peer: Collaboration and Sharing over the Internet, Boston 2002.

[8] Liquidnet, http://www.liquidnet.com, 2004.

[9] Loguinov,D., Kumar, A., Rai, V., Ganesh,S.: Graph-Theoretic Analysis of Structured Peer-to-Peer Systems: Routing Distances and Fault Resilience, http://www.acm.org/sigcomm/sigcomm2003/papers/p395-loguinov.pdf.

[10] Lohmann, M. /Nopper, N. /Henning, P.: Elektronisches Kontrahentenmatching im agentenbasierten Wertpapierhandel, in: Informationssysteme in der Finanzwirtschaft, Weinhardt, Christof /Meyer zu Selhausen, Hermann /Morlock, Martin (Ed.), Berlin, Heidelberg, pp.269-284.

[11] Maxemchuk, N. F. /Shur, D. H.: An Internet Multicast System for the Stock, in: ACM Transactions on Computer Systems, vol. 3(19) 2001, pp. 384-412.

[12] Miller, M.: Discovering P2P, Alameda 2001, p. 91 – 113.

[13] Rowstron, A., Druschel,P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proc. IFIP/ACM Middleware 2001, Heidelberg, Germany, November 2001.

[14] Schmidt, D. /Vinoski, S.: Distributed Callbacks and Decoupled Communication in CORBA, in: The C++ report: The international authority on C++ development, vol. 9(8) 1996, pp. 48-56.

[15] Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications, in: Applications, Technologies, Architectures, and Protocols for Computer Communication Proceedings of the 2001 conference on Applications,

technologies, architectures, and protocols for computer communications, San Diego, 2001, p. 149 – 160.

[16] Zhao, B., Kubiatowicz, J., Joseph, A.: Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing, URL: http://citeseer.ist.psu.edu/rd/ 26183517%2C491241%2C1%2C0.25 %2CDownload/http://citeseer.ist.psu.edu/cache/papers/cs/23847/http:zSzzSzoceanstore.cs.berkeley.eduzSzpublicationsz SzpaperszSzpostscriptzSztapestry_sigcomm_tr.pdf /zhao01tapestry.pdf, 2001.