Hierarchical Loop-star Topology in a New Internet Auction Model

Mohammad-Reza Khayyambashi [1], Behrang Barekatain [2]

Abstract**—**Internet auction has given another new meaning to Internet and a suitable way for achieving the most important man's life purpose which is beneficial trade in a simple and easy way. Internet auction is a special way in the field of supply chain management for running real auctions and their samples in electronic way without time and place limitation. It is obvious that the basic elements in an Internet auction can be beneficial for the supply chain management even when there is no need to have an auction. Today there are different ways of Internet auction. Two phases of Internet auction are one from user's point of view who expects to find items in the easiest and cheapest way and purchase it and the other phase is the technical view which is the relationship between Internet auction servers. They not only provide the fast and reliable services for the users, but also they should maintain transparency condition. The main aim of this article is to present an effective, efficient, scalable and suitable algorithm for the mentioned purposes.

Keywords**—**AS (Auction Server), CAS (Centre Auction Server), MAS (Main Auction Server)**,** BS (Backup Server)

I. INTRODUCTION

THE purpose of this study is presenting a suitable model by using loop-star topology for connecting Internet auction servers with each other and aims to develop Internet auction and perform the supply chain management with more ease and care. It also aims to show that the proposed model is more efficient than the existing models. This efficiency is mentioned at the end of the article and it shows that this topology demonstrates a good respond to the client's demands.

Nowadays, Customers are interested in using the technology of online business, they like to enter a virtual store and use online catalogues for choosing the item they need and pay for it by one of the online paying methods [1]. Meanwhile there will be a meaningful relationship between seller and customer which results a satisfying trade, especially when it is in the form of an auction. Internet auction is one of the most exciting environments in the Internet, because people can buy their needed items in the simplest and easiest way without any need of physical presence at the place of auction in a short time [2]. Today there are several ways of doing Internet auction. The most important ones are as follows:

English Auction: One of the most commonly used type of auction is English auction [3]. In this method all participants try to offer the highest price. At first it starts with a base price then participants offer different bids based on what they can afford. Finally the client with the highest price will win the auction [4].

Sealed bid Auction: At this method buyers offer their prices during a limited time. These prices will be kept secret by an auctioneer until the deadline of offering the prices. After checking all the prices the winner will be announced [3].

Single Round Sealed Auction: In this method there is no competition. Despite the English method that buyers try to offer the highest price, all participants offer their price simultaneously and seller chooses the best price [3].

Multi Round sealed Auction: In this method all the participants should offer their prices during a limited and announced time. At the end, the winner will be chosen and its price will be announced to others. Then another auction will be started soon after, with its own limited time. The procedure is like the single method [3].

Dutch auction: This method is useful for the situations in which seller should sell the goods and items as soon as possible (like retuning items). In this method, unlike the English method, at first the auctioneer starts the auction with the highest price and then depending on the conditions decreases the base price. The auction will finish when the definite time is finished, all the items sold or the price decreases to less than the least permitted price [3].

Sometimes there is a mixture of all of the five methods, but certainly all the methods for Internet auction should meet the following criteria:

Confidential level:

In an Internet auction the amount and limits for presenting information should be clear and mentioned. Sometimes we even do not like to announce the offered price by winner.

Recent advances in information and communication Technology have facilitated dematerialization of purchasing procedures using electronic auctions. Besides price-only auction, multi-attribute auctions [7], [9], [10] allow

[1] Department of Computer Engineering, Faculty of Engineering, University of Isfahan, Isfahan, Iran ,
e-mail: M.R.Khayyambashi@eng.ui.ac.ir
[2] M.Sc. in Computer Engineering, Isfahan, Iran,
e-mail: Behrang_Barekatain@yahoo.com

negotiating on multiple features, involving not only the price, but also other attributes such as quality, delivery terms and conditions.

All the rules for beginning and finishing an auction should be mentioned completely definite. In order to prevent any possible doubt or problem during the action, these rules include the relationship between the clients and the servers and also servers with each other.

Limits in an auction:

Basically in each auction there is a minimum price and the offered prices can't be lower than it. Also the limited and definite time and the validity of offers should be rule governed. There are some key points in an Internet auction all of which together will complete the auction process. These points are as follows:

Client: The system by which the buyer announces its price in an auction;

Server: In an Internet auction there are one or more servers that run and manage the auction (auctioneers). In fact the guarantee for performing and following the rules of auction including the three ones above is these servers duty. In this specific field we call them auction servers;

Rules: Rules are those instructions and laws that tell how the clients and servers should interact with each other and also include rules about server interactions;

Connection structures between the elements: This section includes paths, hardware and software, equipments, relational protocols and especially network topology.

Reviewing all the above points we understand that the most important point is the connectional structure between the elements. Hardware is the main point in the first and the second methods which are considerable in point of its costs, as today hardware and software can be connected with each other very easy. Also the rules depend on the way of running the auction which should be designated by the auctioneer. The important matter is structural connections between the parts in an Internet auction. This can be viewed based on the topology framework and the way of protocol interaction between the parts. The purpose of this study is introducing a structural topology which can be used as a suitable network structure in an Internet auction besides having outstanding capabilities.

II. DIFFERENT VIEWS IN AN INTERNET AUCTION

In an Internet auction there are some key points which may be hidden from viewers. But if there is not a good management, it may cause confusion and complaints among participants. Managing customer's offers and requests in a quick and correct way and finally acceptance or rejection of that offer seems a simple task. But in fact it has a lot of technical complications which need suitable management in order to prevent or resolve the difficulties and problems and to increase the efficiency of system and finally perform a successful auction without any unpleasant outcome. So the need of having a trustworthy and reliable system for exchanging information especially when this is related to exchanging money is very important and necessary.

In addition, we should also notice the prices comparison which is followed by moral commitments. There are two overall views to implement connection substructure in an Internet auction: Central and Distributed [6].

Central view:

In this method all the participants send their requests to a central server and this central sever is responsible for comparing, accepting or rejecting the prices. Also this server is responsible for controlling the rules of the auction. The client offers its price to the central server and then it will be put in the queue and checked. If the offer is less than or equal to the last minimum price a "*Fail*" message will be sent to the client. Otherwise the "*Accept*" message will be sent to inform the client of the acceptance of its offer. After accepting the last offer the server waits for a while and if a better offer was not offered the winner will be announced and the auction finishes. The procedure is shown in Fig. 1.

.

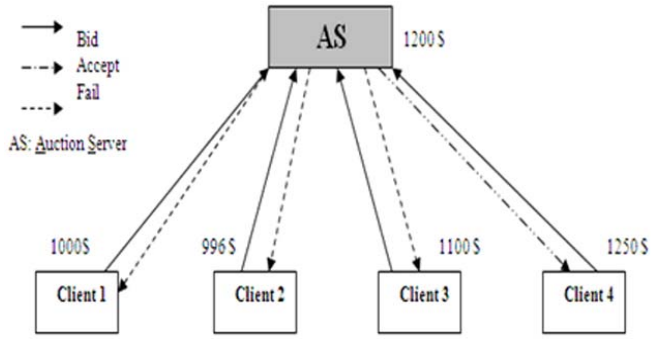

Fig. 1: Running the central view bid, stands for the sent offer from the client

This method has its own advantages and disadvantages. Some of the advantages are as follow:

- Easy management of offers due to a central procedure;
- Simplicity in designing the structures and connections;
- Lower costs in running the system.

And some of the disadvantages are as follow:

- Decrease fault–tolerance: The whole system will have a low fault-tolerance because of the high possibility of breakdown in the system which can lead to the auction being unavailable. In fact in this method we just have a central controller that its failure will lead to the system's failure[1];

[1] Point of Failure

- The long time for responding to offers because of the huge number of unprocessed offers in the queue of server;
- Difficulty in development;
- This system lacks fairness.

Considering the problems of the first method, switching to another method is proposed, in which some servers cooperate together and serve the clients' requests in an Internet auction. Using this new method leads to the creation of another method called the Distributed method.

Distributed view:

In this method we use different servers in cooperation with each other to respond to the requests. When a client offers a price, server compares it with the highest price received. If it was less or equal, naturally it can't be acceptable, but if the offered price is more than the highest price still the server can't send the accept message for the client since in another server a higher price may be registered. It is concluded that if the server wants a higher price to give a certain reply to the client; it should have connections to other servers and also to CAS (Centre Auction Server) in order to have a correct comparison. CAS is the supervisor server and keeps the best offers moment by moment. Fig. 2 shows the way of running and performing of this method. This method has some advantages that compensate the short coming of the central method:

Failure of one AS does not cause the failure of whole system. The client, after a specific time, sends its offer to another AS;

Because the offers are distributed among the servers there is no overload of the offers in one server's queue;

It is more scalable.

The main problem which we still face in this method is the presence of a point of failure called CAS.

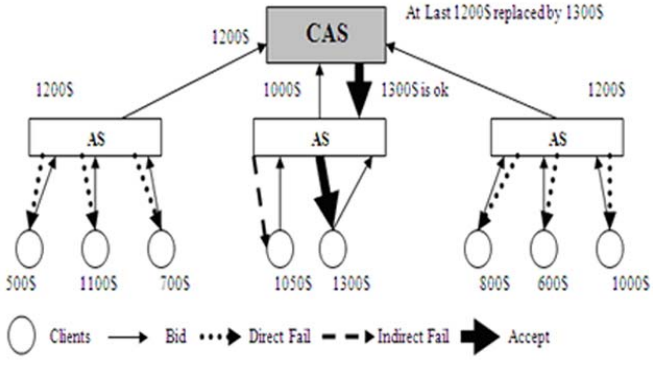

Fig. 2: Implementation of Distributed view

Beside the motioned problem, this method has other points to consider. If we can solve those problems it can be say that we have a fairly certain and complete system in which the auction can be run with fewer risks and more rule-governed. These points can be discussed as follows:

According to what have been mentioned, the servers should be connected with each other. So there need to be some policies for connecting them together. It can be seen in the network topology that how the servers can be connecting for interacting and changing the information. So one of the most important points here is the presence of a well built topology between the servers which in its absence there will be more complicated problems than the central method.

Another point is the cost of creating a method in which the failure of CAS does not cause the failure of the whole system.

To solve the second problem we suggest a backup Server[2] which act as a supporter for CAS, it keeps and serves all the saved information. In fact BS equalizes its information with CAS during the process of replication. This consistency can be done by a reliable protocol [5]. Replication is a technique that helps to reduce access latency and network bandwidth utilization. Replication also increases data availability thereby enhancing system reliability [11]. Also a storage controller can perform backup processing without proceeding via a server and without the need for the server to grasp the state of pair volumes. When an instruction for backup is given from the server while data is being copied from a source volume to a target volume, the storage controller transfers the data to a backup device using differential information between the pair volumes after waiting until data about a range to be backed up and instructed by the server is copied into the target volume [12].

There are two techniques for running replication:

A. Active replication:

In this technique all the servers participating in the process of replication should reply the client's requests. In other words, replicas reply independently and run all–replica method in which all should reply. But like all the other methods this method has some disadvantages too. Some of its disadvantages are as follows:

(i) The resources needed for running the process and the procedure is considerable.

(ii) All the offers should be deterministic.

Deterministic means that the result of a process completely depends on the primary status of the server. In other words, the results of one process completely depend on the primary inputs. So the multi thread servers usually use non-deterministic situations.

B. Passive replication:

In this method one of the replicas acts as the primary server. In fact this server is responsible for receiving requests from clients and responding to them. The backup server (BS) is only related to the primary backup and receives the most up

[2] BS

to date the latest states. In this method there are fewer problems in comparison with the active method. There is no need for determinism in processing the requests. Fig. 3 shows the way of running these two techniques.

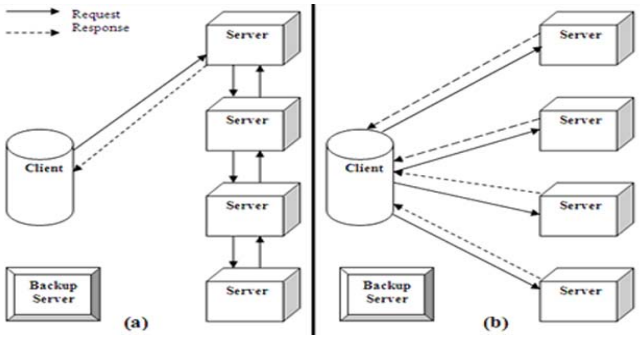

Fig. 3: (a) Implementation of passive technique (b) Implementation of active technique

Considering the points mentioned above, we are going to introduce a suitable topology that uses passive replication and distributed method. In fact using the passive method especially in distributed environments will have some advantages and disadvantages are as follow:

Since there is no need for clients to connect to different servers at once, designing the needed software for clients is easier and clients shouldn't work several systems.

The possibility of changing the procedure in this method is easier, as changing the procedure, for a server is easier than changing it for several clients. (By procedure we mean the way of connections between the clients and servers).

As a sample of disadvantages of the long distances between server and clients may cause some problems such as time out which may causes failure in the whole system [5].

So there are other problems in real environments that there is no a complete solution for them. For example sometimes bidders make some difficulties for the auction. Experimental evidence indicates that efficiency and revenue may be reduced when bidders hesitate to incorporate synergy values into their bids for fear that they will end up winning only part of a desired package [8].

III. INTRODUCING THE PROPOSED MODEL IN THE FRAME OF LOOP-STAR TOPOLOGY

From what has been said up to now, it is concluded that using distributed method accompanied by passive replication is a suitable choice and gives us the best performance. For a clear explanation, we consider the English method to simulate our topology. Surely, all of the mentioned methods can be used instead of the English method in our topology (Fig. 4 shows the overall structure of topology). The reason for using star is the innate capacity of having a high fault tolerance in network's fails. Before talking about this structure it should be mentioned that each star as a group has a CAS which is responsible for that group and one of the CASs as a Main Auction Server or MAS (Main Auction Server) supervises all the CASs in the whole topology.

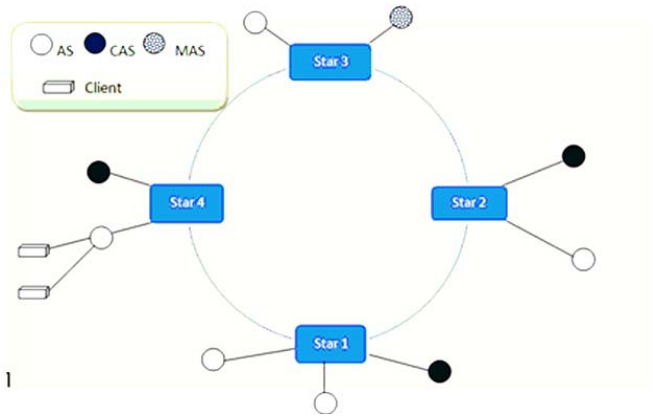

Fig. 4: Structure of the proposed topology

In Fig. 4 there is a loop topology in which each node in it is a star topology and each star is known as a group which contains many ASs one of which is the supervisor called CAS. Each AS serves some clients. By system it means the collection of all the parts and their connections in Fig. 4. The connecting between clients, ASs and CASs is as follows; the client offers a price to the AS to which it is connected. Then this offer will be checked in AS and if it is less than or equal to the highest offered price which is saved in this AS, the client will receive a fail message from it's AS. Otherwise, AS will request the last registered offer in it by sending a request to CAS. In CAS too, two things may happens: if the offer sent by AS is less than or equal to the last registered offer in the CAS, AS will receive a fail message and finally this message will be sent to the client. Otherwise this offer is sent to the MAS. If it is less than or equal to the last registered offer in the MAS, the client will receive a fail message from AS and if not it will get an acceptance massage.

Receiving an acceptance message in MAS to CAS and then to AS will lead to the updating of the last state in them. MAS, CAS and AS are updated, because the client's offer was higher than their last registered offer. It is also obvious a better offer may be registered in MAS. In that case the new client will be the winner of the auction. Fig. 5 shows the pseudo code procedure.

```
send Client.suggest     // Client send its suggest to AS
if Client.suggest <= AS.lastprice then
    send (AS.Client, Fail-Message)
else
    update AS.lastprice
    send (CAS, Client.suggest)     //perhaps CAS's price
                                   is higher than Client's suggest
 if Cleint.suggest <= CAS.lastprice then
     send (CAS.AS.Client, Fail-Message)
else
```

```
    update CAS.lastprice
    send (MAS, Client.suggest) //perhaps MAS's price
                          is higher than Client's suggest
if Client.suggest <= MAS.lastprice then
    send (MAS.CAS.AS.Client, Fail-Message)
else
 update MAS.lastprice
 send(MAS.CAS.AS.Client, Accept-Message) //Client's
                                  suggest is the highest
```

Fig. 5: Pseudo code for communication between clients and servers

The way of selecting the CAS in a group and the MAS of the whole system is an important task which can increase or decrease the efficiency of the system. So having an election method is necessary for selecting a new CAS and MAS if they fail. Methods as follows are proposed for this purpose. In each group, one of the ASs will be chosen as CAS based on the priority number which it is designated to. Then all the ASs will be connected to this CAS and will accept it as their supervisor. A problem occurs when the active CAS can not serve the necessary normal services. In this situation we need to consider an election. For a CAS election we propose two methods:

I. Send message priority–based method:

In this method each AS has a special priority number which can be changed. This priority is based on some parameters such as hardware strength, type of operating system and other properties of this kind which can be designated to each AS. If an AS needs to be connected to its CAS, it sends a message to CAS and it will wait for a period of time. After trying for N times, if it does not receive an answer, it will start the election. AS will create a priority list in its memory and will put its priority number in it. Then it will send priority list as ECAS[3] message to all the ASs of its group. Each AS in that group gives two different reactions after it receives the ECAS message:

If this is the first time that this AS receives an ECAS message, it will create a priority list in its memory. Then AS puts its priority number in it as well as the priority number of the sender AS which is in the ECAS message.

If this AS has created its priority list previously, then it only adds its priority number to the list.

After performing each of these steps, AS sends an ECAS message containing its priority number and waits for the result of the election.

Finally after passing 2T, each AS will have a table including all the priorities of the ASs of its group beside its priority number.

T is the needed time for sending an ECAS message and receiving it in another AS. It is suggested that each AS waits for about $2T+\alpha$[4] after sending its ECAS message to ensure all the ECAS from the other ASs in its group have been received. Finally each AS will elaborate its priority list and will find the highest priority. Surely, one of the ASs which has the higher priority than other ASs, will find itself as the highest priority. Finally all the nodes that elaborate their lists will send a message as a RECAS to the AS that has the highest priority and will then complete their investigation. The AS that has found itself as the highest priority in its priority list, will ensure that it has the highest priority and introduces itself as the CAS of group by sending a SECAS[5] message to all other ASs when it receives at least one RECAS message. It is necessary to pay attention to these tips in this method:

No two ASs in a group have the similar priority number.

If a failed CAS becomes active again, it can begin the election process.

At the end of the CAS election process, this CAS will run the procedure of selecting the new MAS of the system, because the necessary conditions maybe made available for it to be the MAS of the system.

Each new AS that enters a group will start the process of selecting a new CAS and if it is selected as the CAS at the end of the process it will start the procedure of selecting the new MAS of the system.

During the election of a new CAS in a group, if an AS needs to connect to its CAS it will wait until the end of the new CAS election and buffers the offers of clients.

The advantage of this method is that if any AS with whatever priority is added to the group, there is no need to inform the other nodes of its presence. The new AS may have the chance of being the new CAS of that group by performing the election algorithm besides introducing itself to the group. This method can guarantee the scalability of topology which is necessary in an Internet auction. Fig. 6 shows the related pseudo code.

```
//When as AS creates, Administrator assigns to it a priority
number which can change
 i=0
 try n
   send (AS,CAS,Request) // AS send a request to its
                                                    CAS
   wait(AS,T)   // AS wait for CAS's reply for T ms
    i=i+1
    if   i>n then do priority.election(AS) // if AS try for
                                  N times and didn't get reply
        call election procedure
 end try
```

[3] Election Central Auction Server

[4] $2T+\alpha$ includes the send time of an ECAS and the receive time of its reply. For having more reliability we add α to $2T$. ($\alpha \ll T$). The symbol "<<" indicates that α is a small value of T.

[5] Set ECAS

```
 send (AS.Client,Reply) //AS get CAS's Fail/Accept
                          reply and send it to Client
Procedure priority.election(ASj)
  begin
    create plist[j] for beginner ASj //AS which call this
                          routine creates its priority list
    Plist[j].item=ASj.proirity-number // ASj add its
                          priority number to  its list
    send (plist[j],ECAS,ALL)  // ASj send its list and
       ECAS(contain its name) to all  AS in its STAR
    do  wait_RECAS(ASj,2*T + a)
   for each ASi in this STAR
      get (ASi,ECAS,plist[j])
      if plist[i] not exist then
          create plist[i]
      end if
      copy (plist[i],plist[j])  // ASi copy ASj's list in its
                          list
      Plist[i].item=ASi.proirity-number //ASi add its
                          priority number to its list
      send (plist[i],ECAS,ALL)  //ASi send its list to
                          all other AS
       do  wait_RECAS(ASi,2*T +a)
    end for
end

Procedure wait_RECAS(ASk)
   begin
     if  (2*T+a) is expired for ASk then
       search (plist[k],highest_proirity,ASp) // search
          highest priority number in plist[k] and put its
                          name into ASp
       if ASk=ASp and (ASk receives a RECAS from
                          another AS) then
          send (SECAS,ASp,ALL) // ASp send a
                          SECAS to all
          which indicates that ASp is the new CAS in
                          this STAR
  end
```

Fig. 6: Pseudo code for election of a CAS by Message priority-Base method

II. Send message response–based method:

This method is like the previous method with the difference that instead of sending its priority number, the ASs will send the number of their response messages to that point. The response message is the number of all of the offers which the AS received from its clients and replied with accept or fail . At the beginning of an AS's activity, the number of response message is zero. The new CAS is an AS which has the most response messages since the start of its activity. It seems that the response message parameter is better than priority number parameter. It is obvious that if a new AS enters the group there won't be any election process, because the value of the response messages is zero for it. But if an inactive AC becomes active again, it can begin the election procedure of the new CAS. The number of response messages can save in the stable storage of the AS and AS will access it after it becomes active again. The pseudo code for this method is like the previous one. However instead of priority number we put the number of response messages in the list.

All of the previous methods were for electing a CAS in a special group. For selecting MAS which is in fact responsible for managing all the CASs and whole system and has an online relationship to BS, we suggest other methods which will come in the following. For MAS election we propose two methods:

I. Token ring priority–based method:

If in a system we conclude that there is no active MAS, the first CAS which identifies this problem will send a message to all the other CASs in other groups to inform the entire status and take the system in block state. Naturally, there is no service for clients and a time out will occur on the clients' side, so they will send their offers again. The block state will continue until the end of an election for a new MAS. In this method the CAS that identified the problem which had caused the system breakdown and MAS destruction will start the election process. This CAS will make a packet called an EMAS[6] token and will put its priority number, network address such as IP address and the name of its group in it and sends token to the next group. In each group the CAS will do the same. If in a group the CAS receives the token with its own identifications, it means that the token has completed a complete circle. At this time the CAS will check the token and find the CAS with the highest priority number. Then it will send a message to that CAS (if it is not itself) and to all the other CASs to introduce a new MAS. Finally the new MAS sends the ERMAS[7] message to all other CASs and releases the system from its block state in addition to introducing itself as the new MAS of the system. When the token arrives in the group of the previous MAS, two things may happen:

A new CAS is active in this group, so it puts its identifications into the token and sends it to the next group.

A CAS has not been chosen in this group yet. In this situation the token will be sent to the next group by one of the ASs without anything having been added to it . In this group the CAS which will be selected in the future will begin the procedure of electing the new MAS to evaluate its chances of being MAS.

The token circulation is just in one direction. The pseudo code of this method is shown in Fig. 7.

```
i=0
```

[6] Election MAS
[7] Election Release MAS

```
try n
  send (CAS,MAS,Request)  // CAS send a request to
                                        its MAS
  wait(CAS,T) // CAS wait for MAS's reply for T ms
   i=i+1
   if  i>n then
        do priority.election(CAS)  // if CAS try for n
   times and didn't get reply call election procedure
end try
send (CAS.AS.Client,Reply) //CAS get MAS's
        Fail/Accept reply and send it to Client
Procedure priority.election(CASj)
  begin
    send (Block_message,ALL)   // send message to all
                                    CAS to block system
    create EMAS.Token(proirity,name)
    add.Item.EMAS.Token(CASj.priority-
     number,CASj.name,CASj.address)
     repeat
        send (EMAS.Token, next_star,CASk) // send
                            Token to next star's CAS
        add.Item.EMAS.Token(CASk.priority-
         number,CASk.name,CASk.address)
     until (CASj get EMAS token contains its priority-
            number and name)
     search(EMAS.Token,highest_proirity,CASp) //put
             the Highest priority CAS name into CASp
    send (ERMAS,CASp,ALL) // CASj send a message
          to all  other CAS which  indicates that CASp in
          the new  MAS
    send (Unblock_message,ALL) // CASj send a
             message to all  to unblock the system
```

Fig. 7: Pseudo code for MAS election base on Token-ring priority-base

II. Token ring response–based method:

This method is just like the previous one, but in this method each CAS instead of putting its priority number will put the number of response messages it has received to that point into the token and finally the CAS with the greatest response message numbers will be known as the new MAS. It is again said that the response messages number parameter is more suitable than the priority number parameter. The pseudo code of this method is like the previous method but instead of priority numbers we put the number of response messages.

Considering all the points mentioned above, the connections between the ASs and the CAS in a group and the CASs with the MAS of the system can be like the connections between some of the ASs in a group with the CAS responsible for that group and all the CASs with the MAS which is working as a CAS in a group. In this method no AS is directly connected to MAS, unless that MAS is as the CAS of that group which the AS is a member of. Logically, all of the ASs of that group know the MAS as their CAS. Since there is a possibility that the active MAS may breakdown we face two situations:

-Choosing MAS that the algorithm of which has been processed;

-Transferring the latest status of the new system to the new MAS;

To solve the second problem we suggest providing an online backup server which is always connected to the active MAS. As soon as the new MAS is activated, it will connect to BS and will ask for the last status of the system. Then it will compare it with its registered information and if its information is more recent, it will transfer it to the BS[8]. Otherwise, the information from the BS will transfer to the MAS. The connection between the MAS and the BS is online. The online method costs more than the offline method, but since there is not too much information there will be no problem. It should be mentioned that the BS can register the status of the CASs of the groups and send it to the new CAS when the new CAS needs it.

The connection between clients and servers should be in a way that clients aren't confused, do not face a complex system and get their response very soon and the failure of a server is not felt by the clients. Considering the distributed structure in proposed topology all the offers will not end up in one system. They will be processed in the three levels of AS, CAS and MAS and in each level all the offers will not be send to upper levels. Since there are a lot of stars (or groups) with a lot of ASs in them, the traffic in high load in comparison with other topologies will decrease considerably. If we suppose:

N: as the number of all the offers from all Clients;

P: as the number of all the stars in the network;

M: as the average of the number of ASs in each star;

Then the number of all the offers that each star receives is U = N/P and the number of offers which each AS responds to is R=U/M. This recommended method shows how the load of the system will be broken. From R requests for each AS, X request to CAS (X<<R) and Y requests to MAS (Y << Z) will be transferred. This attempt is for the accept or reject response to be given in the AS. Therefore the system will have a high throughput.

In the system each client can be easily connected to the nearest star and the distance is not an agent for increasing the response time of the system and finally time out will not happen much in the clients. The client will receive the response very soon from AS. In an AS, only the offers which are higher than the price of AS itself can go to the upper level. Then the network traffic will decrease a lot and the speed of responding will be the ideal in the proposed topology and finally the system will seem ideal for users. Considering the following reasons, the proposed topology will keep its fixed

[8] Because, it is possible that new MAS has a new Bid that yet it has not registered in the BS.

status when a server fails.

If an AS runs into trouble, the client can switch into another AS in that star and there is no need to connect to an AS in another star.

If CAS or MAS runs into a problem, by using the algorithms introduced in this article and by the BS, the system will maintain its fixed status.

Because of the presence of the star topology in the proposed topology, it is possible to add an AS to the system easily. Also it is possible to add a group. This capability is an innate advantage of the star topology that can be easily created and developed.

IV. COMPARISON OF NEW TOPOLOGY WITH TRADITIONAL AND USUAL TOPOLOGISE

One of the most famous and common topologies is Tree. So we compare it with the proposed topology with it and then with other topologies. Notice these instances in this comparison:

In the proposed topology the maximum number of levels that clients should pass to reach the MAS and compare its bid is three. It means the state that the client is in the star where the MAS is not present. At this time the client follows its request in the following manner:

-Sending bid to its group's AS;
-Sending bid from AS to CAS of its star;
-Sending bid from CAS to MAS in another star.

We should pay attention that in the tree topology the MAS is always in the root. So if we assume that the number of levels that should be passed to reach the root is equal to the number of tree levels, then clients should pass the number of tree levels to reach the MAS. However, the maximum number of levels for passing is limited to three levels in the proposed topology. As the number of ASs increases in the tree structure, the number of tree levels will increase too. Now if we suppose that the distance of system is not so important in meter and the number of connecting levels is considered as measuring criteria, then for the proposed topology in this article the most connecting distance for comparing one bid in one sending and receiving from client to MAS is equal to six (considering following the clients requests as it was mentioned above). This number never increases by increasing of ASs. Because by adding one AS, this system will be placed in an existing star or will be placed in a new star. In both cases the number of levels won't increase. Tree topology will be better than loop-star only if it has one level, because in this case the length of connection equals four (client to AS, AS to root, vice versa). But if the number of levels in the tree is more than two, then in all cases the length of the connection in this method will be better. If the level of the tree is two, the connecting length is equal in both methods. In fact the number of ASs abounds in real environment. For example with two hundred ASs, the number of levels in a completely balanced binary tree is seven (an absolute balanced state is the most optimistic view for tree to decrease the levels) consequently the length of connection between client and MAS in one sending and receiving is equal to 16. It should be considered that the worst state that may occur in the proposed topology (it is when the client is not in the same star that MAS exists) has been compared with the best state in the tree topology. The only thing which remains is that we consider our tree as K-fold of one or two levels. It means that by adding each AS we consider it as one node in the first or second level in tree and the levels of tree should not exceed to more than two levels. It is obvious that in this state the connecting length is equal to the proposed topology. But the simplest problem that may occur in this tree structure is that if the root is damaged and removed, we have a lot of sub-trees with few nodes the functioning of which will not be very useful. It is clear that the closer to a binary tree, we get by offering a suitable bid from one client, the more ASs to the root will be updated and if the root is damaged and out of order we have two sub trees with ASs that most likely have the last offered bids. However, the problem of the number of levels exists in this schema. In the state that the tree has one level then by removing the root we have N independent ASs that have no connections with each other at all. It means rising of an N-node tree to separated N nodes will cause removing the whole tree structure and as a result damaging the connecting topology.

Thus, the loop-star topology which is mentioned in this article can be compared with the famous following topologies.

A. The central server structure:

As it was mentioned, in the case that there is only one AS and a lot of clients, the traffic will increase at the queue of the requests of the servers and overflowing may occur in the servers or time out in the clients. But in the proposed topology the load will be divided between the ASs and the problems will decrease.

B. The pure star structure:

Although the star structure itself increases the efficiency of the system, if we just have one star and the distance between AS and the client or even the distance between two stars increases, we will still have the problem of time out. But in this topology, each client will join the nearest AS and each AS will be part of the nearest star. The loop structure that joins the stars to each other will allow having many stars attach each other. If the number of ASs increases, then it is possible to create a new star and join that AS to it. This can make the topology more scalable than pure star or loop topologies.

C. The pure loop topology:

The presence of one loop can connect several systems to each other. But the increase of systems is limited in a pure loop topology and increasing the number of systems may decrease the efficiency of the system. The structure of the

token ring network based on the loop makes this obvious.

V. EVALUATING THE PROPOSED TOPOLOGY FUNCTIONALITY BY SELF-DESIGNED SIMULATOR

To evaluate the function of the proposed topology a simulator was designed in SUN Java™ Studio Enterprise 8.1 environment and the topology was run under English method (this simulator is available by request). Several different tests were conducted with different characteristics in three general categories that are grouped as follows. The qualifications of performing simulations were:

Using several stars and each star contains several ASs and for each AS several clients that sometimes the clients and ASs have the same qualifications and sometimes have completely different qualifications in each group.

While performing each test, different things were done or changed such as: AS priority, number of clients, activating or deactivating of AS or clients, changing the number of requests in each period of time for each client, the complete deletion of one client from the auction and inserting new AS or client in the system.

The duration of time for each test group was between 25 to 30 minutes.

Sending bid from a client can be performed by one of the following methods:

Automatic bid sending from each client between a minimum and maximum value which is defined at the beginning of the simulations.

In hand bid from each client. In this case we tried to suggest a higher offer than the last.

Our tests were done several times and in different situations by self-designed simulator in three categories which are as follows:

First category: using the method of number of response messages in choosing the CAS or the MAS by using 4-1 qualification.

Second category: Using the priority number in choosing the CAS or the MAS by using 4-1 qualification.

Third category divided to the following two categories which finally their results merge together as an average value:

Using the number of response messages in choosing the CAS or the MAS by using 4-2 qualification.

Using the priority number in choosing the CAS or the MAS by using 4-2 qualification.

We had three Figures as outputs (Figures 8 to 10) that are explained as followed.

A. The Figure of the results of several tests in the first category

These tests were done in several states by 15 ASs in four separate stars in which three stars have 4 ASs and the other star has three ASs. 32 clients were created with the average of 2.1 clients for each AS. All the qualifications of the tests which were mentioned were considered. The results are shown in Fig. 8. The letter "A" in Figures means AS.

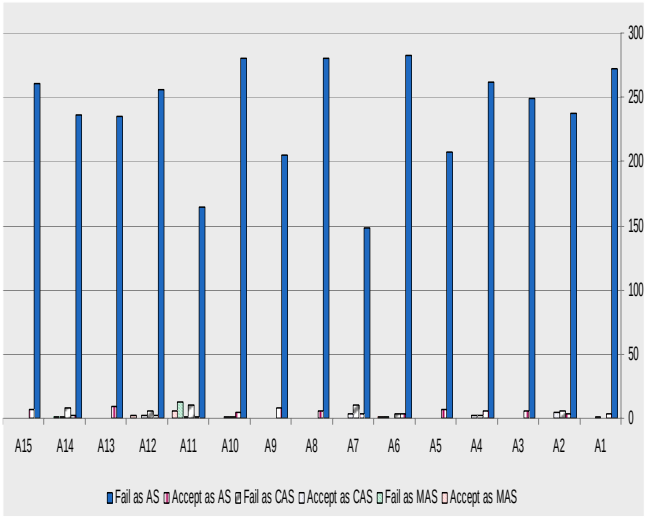

Fig. 8: Graph showing the results from the first category of the simulations

B. The Figure of the results of several tests in the second category

The properties of test were similar to the last test. Here we have 15 auction server and run the second category of tests on them. The result of simulation shows in Fig. 9.

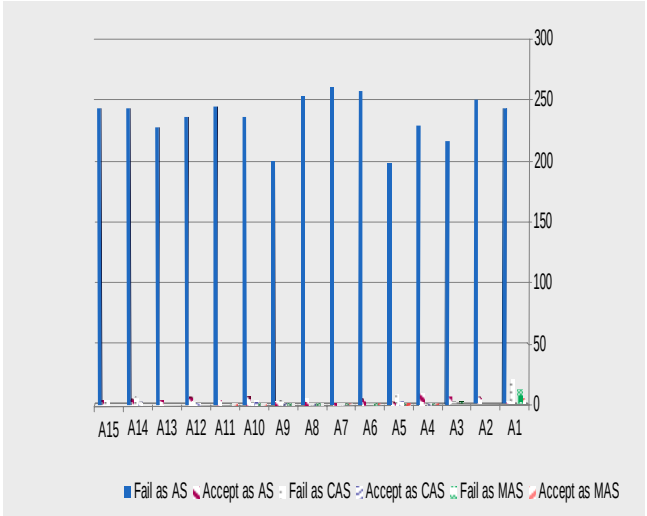

Fig. 9: Graph showing the results from the second category of the simulations

C. The Figure of the results of the average of several tests in the third category

In this category of the tests several different states were run by 4 ASs in two separate stars that in each star there were 2 ASs. 8 clients were defined and for each AS there were 2 clients. All the qualifications of tests were considered. Fig. 10

shows the results.

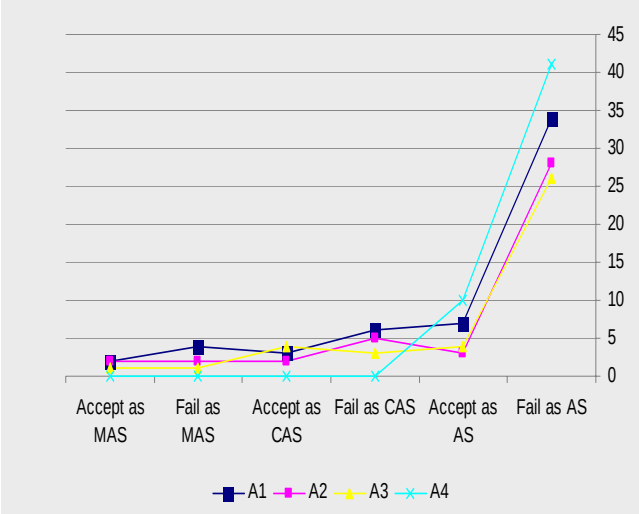

Fig. 10: Graph showing the results from averaging of the third category of the simulations

Considering the Figures these concepts should be explained:

Fail as AS: This shows the number of offers that the AS receives from its clients and since the offer is less or equal to the last registered offer in the AS, it rejects that offer.

Accepted as AS: It shows the number of messages that AS receives from its clients and since the offer is greater than the last registered offer in the AS, it accepts that offer, updates itself and sends the offer to its star's CAS for more evaluation.

Fail as CAS: It shows the number of messages that the CAS receives from the entire ASs in its star and since the offer is less than or equal to other received offers, the CAS rejects it.

Accepted as CAS: It shows the number of messages that the CAS receives from the entire ASs in its star and since the offer is more than the last registered offer in the CAS, it accepts the offer, updates itself and sends the offer for more evaluation to the MAS. Fail as MAS: it shows the number of messages that MAS receives from the entire CASs in the system and since the offer is less than or equal to the last received offer, it rejects it. Accepted as MAS: It shows the number of messages that MAS receives from the entire CASs in the system and since this offer is more than the last offer, it accepts the offer, updates itself and sends the approved registration of the offer to the client.

It is quite clear that in average 96% of all the bids that clients send to each AS will receive the desire response message in that AS and there is no need to send it to the higher servers such as CAS and usually 74% of the sent offers to the CASs are not sent to the MAS in another star and will be answered in the same CAS. Above Figures show that in this proposed topology all the offers will be answered in the lower levels near to the clients without sending to the higher levels. Consequently, we can consider the following advantages for our proposed topology:

Less need to wide bandwidth for the network: The most important problem in providing vast bandwidth is about WAN connections. In this topology the ASs and the CASs are in a LAN that usually has no problem about bandwidth (because of its small scale). The problem of providing the vast bandwidth in the WAN for connecting stars together is solved by decreasing the numbers of the sent messages to the MAS. In other words it is possible to have this topology, with a narrow bandwidth.

Decrease in the response and waiting time in the clients' side: In this topology by its hierarchical structure the clients will receive their fail or accept message in a short time.

Increase of the fault-tolerance in the whole system: In the proposed topology in the case that a CAS or the MAS crashes, the whole system will not face a non-compensation trouble, because the substitute will be replaced as soon as possible and since most of the evaluated offers will be finished in the AS, the system will not have a lot of mistakes in responding the clients. If one AS breakdowns, its clients will move to another AS in that star (or in the other star). So the fault-tolerance of the system in possible break downs is high. If one client offers a higher bid than the last offer in the AS and that AS can't find its CAS, it can buffer the bid and wait for choosing a new CAS. All of these can cause just hundreds of seconds of delay for receiving accept or reject reply for the clients which is not noticeable in an auction.

Absence of a jammed traffic area in the network: Because of a good and suitable distribution of offers, there is no high jammed traffic point in this topology and a fail point which is one of the most important disadvantages in most of the topologies does not exist here.

Easy development without worrying about decrease of the above advantages: Here the system can be developed very easily without worrying about the increasing of the response time or waiting time for the clients, the need of widening the bandwidth or decrease of the fault-tolerance in the system.

Finally for more detailed evaluation of the results, notice to the Figures 11 to 13. In these Figs the average number of offers which each system has received as MAS, CAS or AS is shown.

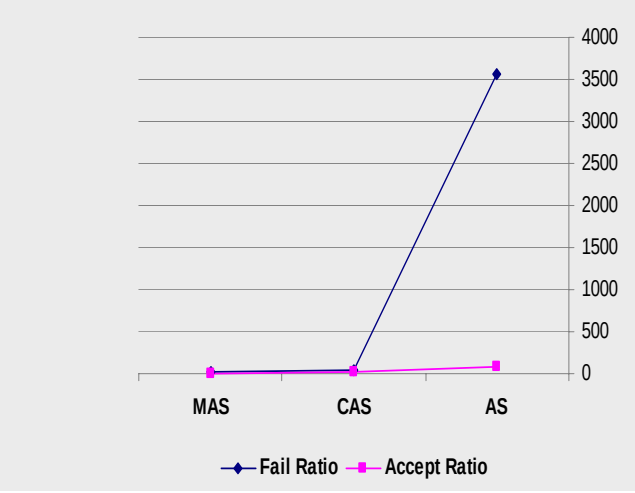

Fig. 11: Graph showing the results from the first category of the simulations

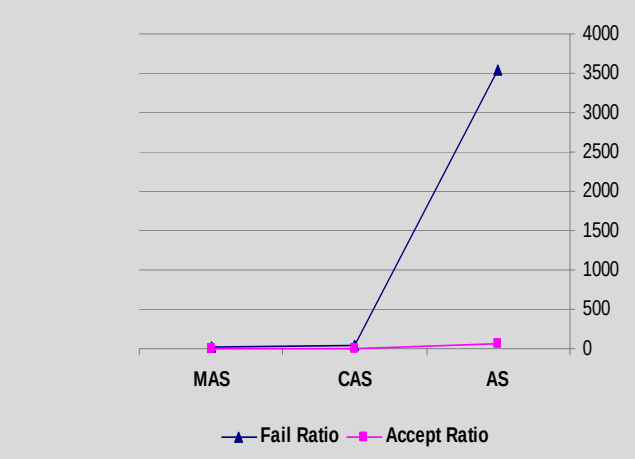

Fig. 12: Graph showing the results from the second category of the simulations

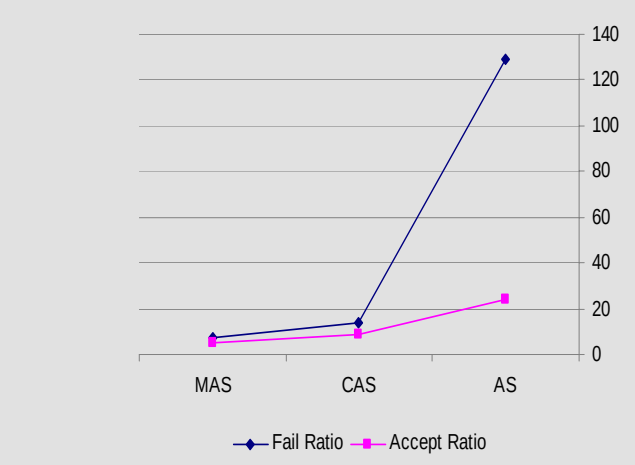

Fig. 13: Graph showing the results from averaging of the third category of the simulations

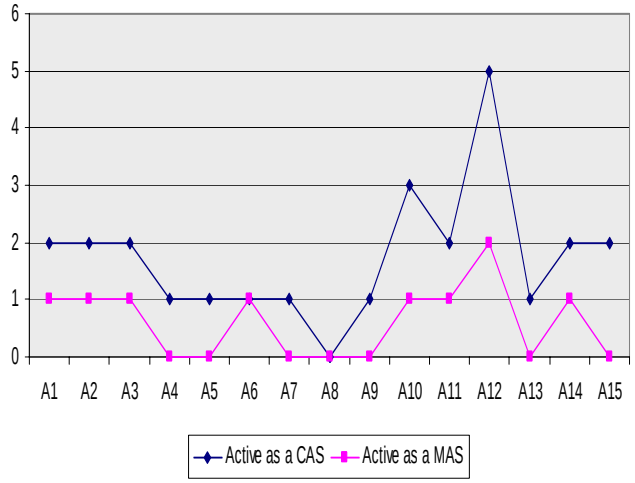

Fig. 14: Graph showing the results from the first category of the simulations

At the end of this section for getting a better perception of dynamic features of this topology, the Figures related to the average times of each AS changes its state to the CAS of its star or the MAS of the whole system are presented (Figures 14 to 16). These changes of the state are based on the first and the second part of the qualifications.

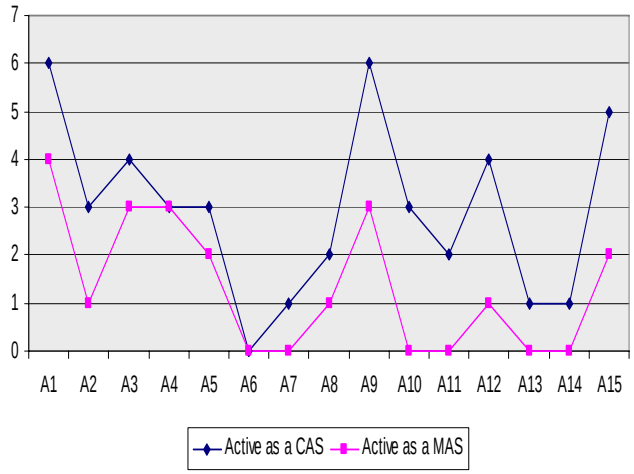

Fig. 15: Graph showing the results from the second category of the simulations

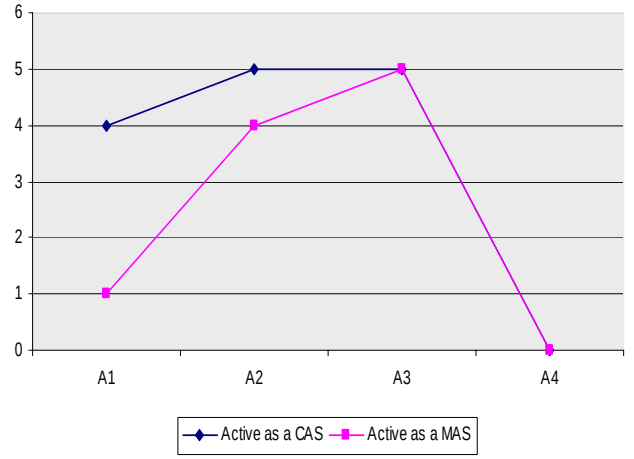

Fig. 16: Graph showing the results from averaging of the third category of the simulations

VI. CONCLUSION

In this article besides evaluating the existing topologies in the Internet auctions, a new model in a hierarchical loop-star topology framework was offered and evaluated completely. This method has created a new schema in the topologies of the Internet auction and has many advantages such as high scalability, fairness, high throughput, low response and waiting time and high fault-tolerance that lead to it being considered as an effective and efficient topology in the Internet auction and finally in supply chain management.

REFERENCES

[1] F.Panzieri and S.K.Shrivastava, "On the Provision of Replicated Internet Auction Services", *In Proceedings of the IEEE Workshop on Electronic Commerce (WELCOM '99), part of the 18th IEEE Symposium on Reliable Distributed Systems (SRDS '99)*,1999.

[2] B. Barekatain, "E-commerce Aspects", *Science and Computer Magazine*, Number 59, 42-45, 2006[3] M, 2006.
[3] Kumar and S.I Feldman. "Internet Auction." *Proc. of the 3rd USENIX workshop on Electronic Commerce*, pp.44-60, 1998.
[4] Makoto Yokoo. "Internet Auctions: Theory and Application." *Journal of Japanese Society for Artificial Intelligence*, Vol. 15, No.3, pp. 404-411, 2000.
[5] R.Guerraoui and A.Schiper, "Fault – tolerance: From Replication Techniques to group communication, "*IEEE computer, vol. 30, pp. 68 – 74. Apr. 1997.*
[6] P Ezhilchelvan,G. Morgan – *International Symposium on Autonomous Decentralize Systems,2001*.
[7] J. E. Teich, H. Wallenius, J. Wallenius, and A. Zaitsev. A multi-attribute e-auction mechanism for procurement: theoretical foundations. *EJOR, 175(1):90.100, 2006.*
[8] Brunner, C., J. K. Goeree, C. A. Holt, and J. O. Ledyard. An Experimental Test of Combinatorial FCC Spectrum Auctions*," Working Paper, Caltech, 2007.*
[9] S. Strecker and S. Seifert. Electronic sourcing with multiattribute auctions. In R. H. Sprague, editor, 37th nt. *Conference HICSS, pages 701.712, Hawaii, USA, 2004*
[10] M. J. Bellosta, S. Kornman, and D. Vanderpooten. A framework for multiple criteria english reverse auctions*. In Int. Conference IAT 2005, pages 633.639, 2005.*
[11] Rashedur M. Rahman , Ken Barker and Reda Alhajj, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada, "Replica Placement Strategies in Data Grid", *Springer Netherlands, Volume 6, Number 1 / March, 2008, 10.1007/s10723-007-9090-8, 103-123, 2007*
[12] Takeshi Ido, Kiichiro Urabe, "Storage system, backup system, and backup method", *Application number 11/375,400, US 2006/0179122 A1, 2006*

Mohammad-Reza Khayyambashi was born in Isfahan, Iran in 1961. He received the B.Sc. degree in Computer Hardware Engineering from Tehran University, Tehran, Iran in 1987. He received his M.Sc. in Computer Architecture from Sharif University of Technology (SUT), Tehran, Iran in 1990. He got his Ph.D. in Computer Engineering, Distributed Systems from University of Newcastle upon Tyne, Newcastle upon Tyne, England in 2006.

He is now working as a lecturer at the Department of Computer, Faculty of Engineering, University of Isfahan, Isfahan, Iran. His research interests include Distributed Systems, Networking, Fault Tolerance and E-Commerce..

Behrang Barekatain was born in Aucken, West Germany in 1974. He received the B.Sc. degree in Computer Software Engineering from NajafAbad University, Isfahan, Iran in 1996. He received his M.Sc. in Computer Software Engineering from NajafAbad University, Isfahan, Iran in 2001.

He is now researcher in Networking, Internet Engineering, Internet Auction, Operating Systems, CISCO Switch/Router Configuration and E-Commerce.