MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Sloan School of Management

15.565 – INTEGRATING INFORMATION SYSTEMS: TECHNOLOGY, STRATEGY, AND ORGANIZATIONAL FACTORS

15.578 – GLOBAL INFORMATION SYSTEMS: COMMUNICATIONS & CONNECTIVITY AMONG INFORMATION SYSTEMS

Spring 2002

Homework Assignment 2 - SAMPLE SOLUTIONS

There are many possible approaches to answering these questions. Some sample solutions are presented here. Some parts are more detailed than we would expect from a typical student solution – they are presented to give you further insights into the questions as well a sense for the range of solutions given.

Question 1 (Strategic Connectivity) *[50 points]*

While account aggregation is one of the hottest services being offered online by financial services firms, some executives are skeptical. Consider, for instance, the following recent news snippet from *ComputerWorld* magazine:

[Banks, brokerages weigh merits of aggregation technology, July 17,2001 (http://www.computerworld.com/storyba/0,4125,NAV47_STO62332,00.html)]

> "An executive at one online brokerage said it would cost his company about $350,000 to install aggregation technology. That's a deal-breaker, considering that the brokerage hasn't heard a clamor for the capability from customers and that it sees no clear promise of a return on the investment, added the executive, who asked not to be named.
> "You're going to adopt something out of fear that others will?" he said. "At the prices [the vendors] want, fear is a bad motivator.""

In another company, which decided to implement the account aggregation technology, Internet project leader, Jonathan Scott, is scared: [Banks See Online Account Aggregation as Necessary Evil, July 23,2001 (http://www.computerworld.com/storyba/0,4125,NAV47_STO62443,00.html)]

> "Scott said he isn't sure exactly how the technology will be received. 'I don't know if it will be an effective tool or not,' he said. And making it easier for online users to compare different money market accounts 'scares me,' Scott added, explaining that the 350,000-customer bank can't match brokerage houses that offer high interest rates."

Meir Shor, CIO at Tel Aviv-based Bank Leumi le-Israel BM, on the other hand, seems to be quite enthusiastic, as reported below: [Israeli bank pushes wireless service despite low usage Sep 3, 2001 (http://www.computerworld.com/itresources/rcstory/0,4167,STO63487_KEY68,00.html)]

> "Despite having a mere 100 users per day for his company's year-old wireless service, Shor, is pushing an IT project that in the next few months will add Web and wireless-based account aggregation and transaction capabilities."

1) Assume you were a manager in one of the above companies (you pick which company), please explain why you would choose to install or not install aggregation technology in your company. Analyze the case using strategic frameworks discussed in class, as appropriate. Please provide your reasons as bullet points with short explanations. *[16 pts]*

Grading on this question are weighted towards the process, rather than the actual recommendation. How well does the student understand the strategic models used in the class.

(Online Brokerage Executive)
Strategic Analysis using the five-forces model:

New Entrants

Account aggregation is likely to attract new entrants such as portals like Yahoo. Although they may not have the advantage of already established trust like the banks, they can present themselves as unbiased third parties who aren't going to cross-sell, thus may use account aggregation as an opportunity to enter into this space. Also, account aggregation presents relatively low barriers to entry because of the nature of information collection from third party resources with a relatively inexpensive software product. Thus the Online Brokerage Company(OBC) faces increased competition in the industry.

Bargaining Power of Customers

Customer's bargaining power will increase with the introduction of account aggregation, since it may help customers shop for lower fees, better interest and mortgage rates. If OBC is behind the curve in terms of fees and rates, then it may lose customers to competitors for not being able to offer as much value as the others. On the other hand, if OBC may differentiate its products from its competitors, so that the comparison is not only based on lower fees or better rates, but on the total value which may be better than the competitors than it may benefit from the increased bargaining power of customers.

New Products and Services

Account aggregation may create opportunities for totally new products and services. OBC may for example offer a service that would optimally allocate customer's capital among different accounts based on specified constraints and objectives. It may also use account aggregation as a platform to cross-sell existing and new financial services products.

Bargaining Power of Suppliers

Account aggregation increases the bargaining power of suppliers in general, technology suppliers in particular. Since OBC has to act quickly to offer aggregation service, it is likely to prefer technology providers such as Yodlee to install the aggregation service quickly, instead of developing the technology in house. Although this may create problems related to trust as banks would then share the customer accounts with these technology providers, they may not have any other option on hand.

Rivalry Among Existing Customers

Customer recruitment and retention

One research report indicated last year that one-third of online bank customers were willing to pay to have all their financial accounts available in one place. By introducing account aggregation, OBC can avoid losing customers to competitors who are likely to introduce account aggregation. In addition, the brokerage may increase its customer base, by recruiting customers from institutions who fail to act as fast.

Based on the analysis above, offering account aggregation and developing new products and services that would enhance product differentiation and possibly spawn new business, seems to be

the only option since OBC faces the increasing threats coming from new entrants, competitors, and increased bargaining power of buyers and suppliers.

2) Assume that it is year 2005 now, and all banks are providing account aggregation services to their customers. Having made the brightest decision to install online account aggregation technology several years ago, you are now realizing that your first mover advantage has subsided and the company managers are going to ask you to make another bright move soon. Luckily, you have taken 15.565/15.578 while you were at Sloan, you remember that "Online account aggregation is not an end to itself. Rather, aggregation needs to become a tool to understand and add value to customers". Explain at least two new ideas that go beyond simple account aggregation and perhaps also require some new additional aggregations be performed. *[5 pts for each of the two ideas]*

Students who can articulate two non-trivial after aggregation services get full points.

1) Automated "private banker": By analyzing all your financial holdings, as well as personal information – such as risk tolerance, this after-aggregation service could identify better ways to allocate your financials to increase the return within your risk tolerance.

2) Automated trading: Agents that can buy and sell stocks on your behalf by reaching your aggregated sources when necessary, and let you specify rules that would govern the transactions.

3) Tax sensitive asset allocation: This after aggregation service would optimally allocate your resources, while considering the tax implications of your investments.

3) The dot-com boom is over, but you still decided to go for the MIT 50K competition with the idea of building an air fare aggregator. One of your team mates has gone through the Computer Project, therefore knows how to use the screen scraping technology developed at MIT, to extract information from different web sites, so you need not worry about the technical details.

a) Identify and describe some of the sources that you would be aggregating for this application. Why did you pick these sources? *[8 pts]*

Note student must provide some description of the sources.

www.orbitz.com -- Scans more than two billion possibilities for flights and prices
www.expedia.com -- Offers hotels, rental cars as well as flight information and prices
travel.yahoo.com – Provides both travel information as well as information on things to do & see

Why: These are the sources I usually look at when I consider buying tickets. Comprehensive & competitive prices.

b) During the 50K finals, a VC suggested that you are more likely to receive funding if you could identify how you could differentiate your service from the others on the market. Surprised and panicked to hear that there are already others in the market you make a search on Google for flight aggregators. Please describe at least one flight aggregator (the extent of its aggregation, capabilities, etc.) and explain how you are planning to differentiate your service from that competitor to get the VC funding? *[8 pts]*

Creative answers, rather than the well known and already applied differentiation strategies, receive more points.

www.deckchair.com, is a flight aggregator in UK. It is not clear whether they reach to the existing internet aggregators such as yahoo travel, or they have agreements with not publicly available database owners. They provide prices in pounds only. ...

Several of the differentiation techniques are already applied in the existing aggregators, such as bundling the flights with hotels and car rentals, and offering a better value to the customers.

One way of differentiation would be through building a meta-flight-aggregator, that would perform parallel searches on existing aggregators, and include popular sources such as priceline.com. The technology to achieve this kind of aggregation, would require more than screen scraping as sites such as priceline.com responds within a time-window.
Therefore the aggregation would need to include e-mail accounts and ability to specify negotiation rules in dealing with sites such as priceline, etc.

c) Suppose that you decided to include Priceline (www.priceline.com) as one of your sources as part of your differentiation strategy. What kind of difficulties (technical, legal, etc.) do you expect including this source in your service? *[8 pts]*

Grading is based on the reasonability of the suggested difficulty. Example technical difficulties are provided below (legal difficulties are also welcome):

Some example technical difficulties:

1) The aggregation software would not only be concerned about pattern matching, but would also have to understand what the answer from priceline would mean, so that it may increase the bid if necessary.
2) Since priceline requires you to buy the ticket once it matches your price, the mechanics and meaning of price comparison would have to be altered. The aggregator may not simply grab a price from priceline and display it as an alternative.
3) Security becomes an important issue, since the aggregator has to store user's credit card info and supply it to priceline, when necessary.

Try to keep your answers to all of the above parts are short and concise as possible.

Question 2 (Network Protocols) *[50 points]*

Consider the situation in Figure 1 which depicts two PCs connected via an Ethernet network (you do not need to worry about details of Ethernet for the purposes of this question). We want to copy file X from the hard $disk_1$ of PC_1, to the hard $disk_2$ of PC_2.

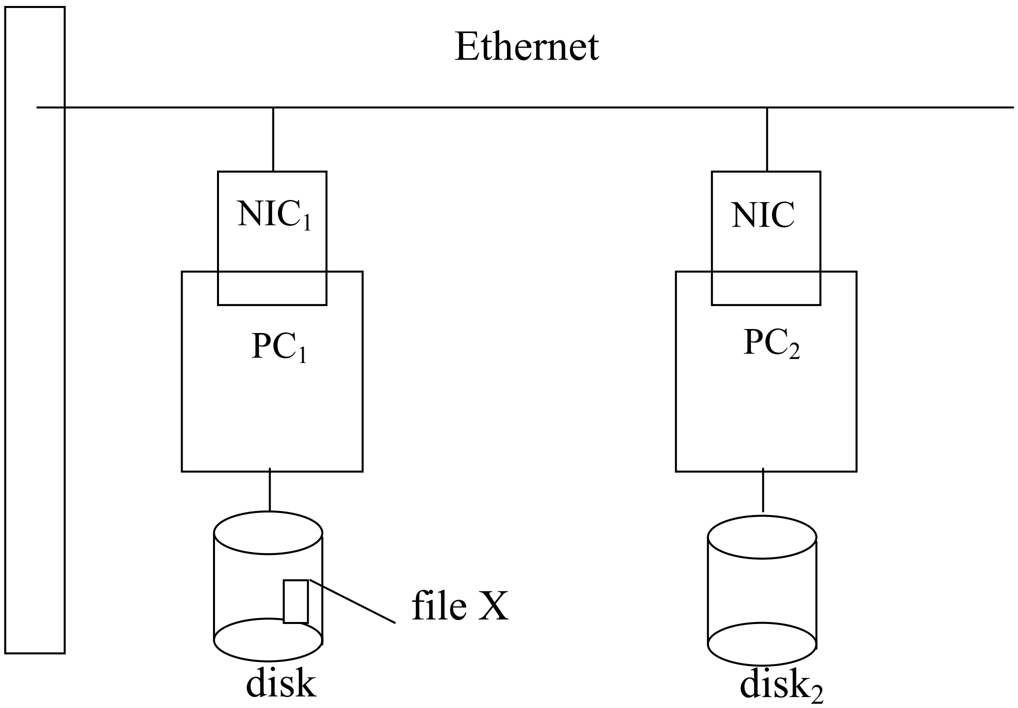

Figure 1. System Configuration

a) You are told that file X is 809,317 bytes in size and that the Ethernet operates at 10M bits per second. Assuming that these are the only 2 PCs on the network, how long would a non-15.578 student expect it to take to copy file X? (i.e., for this part, do not be too fancy but do state any assumptions made). *[4 pts]*

Amount to transfer = 809,317 bytes * 8 bits/byte = 6,474,536 bits. Expected time = 6,474,536 bits / 10,485,760 bits/second = 0.617 seconds

[Approximate answer = 809137/1.25M bytes/second = 0.647 seconds]

The flow chart show in figure 2 depicts our qdftp (Quick & Dirty File Transfer Program) protocols. The NIC (Network Interface Card or, in this case, Ethernet card) acts as the interface and buffer between the PC and the network**.**

b) Identify at least two steps (or actions) that are omitted from the qdftp procedures depicted in Figure 2. Explain briefly what they are and why they would be needed. *[3 pts for each of two steps = 6 pts]*

1. **Initial step to tell PC2 that X is being sent.**
2. **Final step to tell PC2 that entire file was sent, i.e., PC1 is finished.**
3. **Error recovery/retransmit procedure if received packet is not validated ok.**

Despite the deficiencies noted in (b) above, we will assume that Figure 2 is OK for purposes of this question. Table 1 provides information on the speed of various activities and overhead/protocol data that is added at various stages.

c) Using the information from Table 1, annotate Figure 2 showing the amount of time each step takes. Clearly show your calculations and state any necessary assumptions. *[10 pts]*

Step time calculations

1. **Disk read. 8,000 usec + (1,474 bytes * 2 usec/byte) = 10,948 usec**
2. **Analyze packet to send. 1,000 usec (given**
3. **Transfer data to NIC1. Bytes to transfer = 1,474 data bytes + 25 bytes of qdftp overhead = 1,499 bytes. Time = 1,500 usec + (1.0 usec/byte * 1,499 bytes) = 2,999 usec.**
4. **Send data from NIC1 to NIC2. Bytes to transfer = 1,474 data bytes + 25 bytes of qdftp overhead + 16 bytes of ethernet card overhead = 1,515 bytes + 8 bits/byte = 12,120 bits. Time = 10 usec + (12,120 bits / 10,485,760 bits/second) = 1,166 usec.**
5. **Qdftp1 waiting for acknowledgement. No additional time (beyond waiting for steps 6-11 to complete).**
6. **Receiving will take 1,156 usec (same time as #4 but without 10 usec startup delay). This takes place concurrently with step #4.**
7. **NIC2 transfers data to qdftp. NIC2 receives 1,515 bytes from NIC1, strips off the 16 bytes of overhead added by NIC1, and transfers the remaining 1,400 bytes to qdftp. This takes 2,999 usec (using the same calculations as in number 3).**
8. **Qdftp validates a received packet. 2,000 usec (given).**
9. **Qdftp writes packet to disk2. First strips the 25 bytes of overhead added by sending qdftp, then writes the original 1,474 data bytes. This takes the same time as the original read in step 1, or 10,948 usec.**
10. **Acknowledgement packet transferred to NIC2. The acknowledgement packet contains 4 data byte plus 25 bytes of qdftp overhead. To transfer this takes 1,500 usec + (1.0 usec/byte * 29 bytes) = 1,529 usec.**
11. **NIC2 sends ACK packet to NIC1. Bytes to transfer = 4 acknowledgement bytes + 25 bytes of qdftp overhead + 16 bytes of ethernet overhead = 45 bytes * 8 bits/byte = 360 bits. Time = 10 usec + 360 bits / 10,485,760 bits/second) = 44 usec.**
12. **NIC1 receives ACK. Same time as step 11 less 10 usec startup, or 34 usec. Occurs concurrently with step 11.**
13. **NIC1 transfers ACK to qdftp. Transfers 29 bytes, takes 1,529 usec as in #10.**
14. **QDFTP checks for file done. 10 usec (given).**

d) Based on your calculations of part (c):

(i) How long will it actually take to copy file X from $disk_1$ to $disk_2$? *[4 pts]*

Transfer will be broken up into packets of 1,474 bytes or less. 809,317 / 1,474 = 549 full-size packets and one final packet of length 91 bytes. The time to transfer a full-size packet is the total of the times given in steps 1 through 4, 7 through 11, and 13 and 14. Steps not included are:

- **a. Step 5, qdftp waits for acknowledgement. When execution hits this point, the packet has been transferred to the receiving machine and the actions of the sending machine will not be on the critical path again until it receives the acknowledgement for the packet it sent.**
- **b. Step 6, NIC2 receives a packet. You can consider NIC2 to be receiving a bit immediately when NIC1 transmits it, and vice versa. In truth, there is some delay (essentially the speed of light over whatever the transmission medium is), but in this case we will ignore it.**
- **c. Step 12, NIC1 receives a packet. Same explanation as above.**

Therefore, the cycle time to send, receive, and write an entire packet is (10,948 + 1,000 + 2,999 + 2,000 + 10,948 + 1,529 + 44 + 1,529 + 10) = 35,172 usec. To send 549 packets then takes 35,172 usec * 549 = 19.3 sec. The final packet will take slightly less than the 35,172 usec time to transmit, so total time will be right around 19.3 seconds.

(ii) How does this compare with your answer in part (a)? *[2 pts]*

0.6 seconds (predicted) vs. 19.3 seconds (actual) – quite a difference, by a factor of more than 30! Part (a) assumed throughput of the full 10,485,760 bps. Here we are seeing only 809,347 * 8 / 19.3 = 335,480 bps throughput, or about 3.2% of capacity.

(iii) Assuming that you had given the answer of part (a) to your boss, how would you explain why the actual result is different? Provide a simple intuitive answer, not all the details of part (c). Highlight the bottleneck(s). *[4 pts]*

The main point is that there are a lot of other things going on to implement the transfer. It is clear that the major bottleneck is not the ethernet, which is sitting idle most of the time. The biggest bottleneck seems to be reading to and writing from disk. It takes about 11 millisec to read one packet from disk times about 550 packets is 6 seconds. The file write also takes about 6 seconds. Thus 12 seconds just to read and write file even if ethernet took 0 time!

Note that even transferring data between qdftp and the NIC takes more time than it takes to transfer the data across the ethernet. While this is not intuitive, for a variety of technical reasons this is a realistic scenario on early-model ethernet cards and older PCs.

e) We want you to design a new iqdftp (<u>improved</u> qdftp) to get better performance.

(i) Propose changes to the qdftp protocol shown in Figure 2. Explain why you think those changes would help. *[8 pts]*

There are a number of ways in which performance could be improved by changing qdftp. In general, full credit was given if you were able to identify just one significant improvement and explain its effect. Basically, there are two strategies to follow: maximizing concurrency and utilizing economies of scale. These are discussed below.

1. Maximize concurrency. Under the current protocol, only one machine is actually "doing" anything at any given time. For instance, after the sending qdftp sends a packet, it sits idle until it receives an acknowledgement. Why not have it go ahead and read the next packet? This would essentially eliminate the write time of the receiving qdftp from the critical path, as the sender is continuing its processing while the receiver writes to disk. Savings in this case is 10,948 usecons/write * 549 writes, or about 6 seconds! In general,it is desirable to find ways to minimize the time any machine is simply kept idling, waiting for some action by the other machine.

2. Consider economies of scale. Some actions have a high fixed penalty every time they are executed, so it is desirable to modify the number of times you execute them. Two examples are accessing the disk and acknowledging the receipt of packets.

For the disk drive, there is a 8,000 microsecond penalty for every byte read. A good strategy is to increase the number of bytes read each time. Note that it is probably not a good idea to assume that you can read the entire file in at once, since files can be quite large and there is no guarantee that it would fit in memory.

Acknowledging packet receipt is another area that could benefit from such a strategy. For instance, is it really necessary to send an acknowledgement for every packet? Instead, we could allow qdftp to send several packets in a row, and allow the receiving program to acknowledge all of them with a single ACK.

With this in mind, we have designed a new iqdftp. It makes 3 main improvements over qdftp, as follows:

1. It reads/writes 15 packets from/to the disk at a time instead of just one.

2. It sends 3 packets before waiting for an acknowledgement, and allows all 3 to be acknowledged with a single ACK.

3. When a ACK is sent, it is sent BEFORE writing the packet to disk instead of after. While this may seem minor, it actually has a significant impact as a simple way of improving concurrency, as it allows the sending iqdftp to begin reading the next block of data while the receiving iqdftp is writing its block.

One seemingly reasonable suggestion that was made in some student papers was to increase the number of bytes sent per packet. While some went a little far by suggesting that the entire file be sent as a single packet (remember that others may want to use the network, as well), it would seem that in some cases a larger packet size might help. While credit was given for such an answer, it is not incorporated in iqdftp because in fact the ethernet protocol restricts the maximum total packet size to 1515 bytes (it may be smaller, but no larger). Qdftp already sends 1474 bytes per

packet + 25 qdftp overhead bytes + 16 ethernet overhead bytes, for a total of 1514 bytes/packet.

(ii) Using your iqdftp, repeat your calculations of parts (c) and (d). Clearly state any assumptions that you had to make in your calculations. *[8 pts]*

Sending side calculations:

S1. Read 15 packets. 8,000 usec + (1,474 bytes/packet * 15 packets * 2 usec/byte) = 5,220 usec.
S2. Analyze one packet. 1,000 usec.
S3. Transfer a packet to NIC1. 2,999 usec (see qdftp step 3).
S4. NIC1 transmits a packet. 1,166 bytes (see qdftp step 4).
S5. Repeats steps S2 to S4 2 more times (3packets per ACK). Each of steps S2 through S4 depend only on the completion of the previous step to begin execution, and S2 can begin again as soon as S4 is done, so there is no idle time in this loop spent waiting for the receiving machine to do something. Therefore, the total time to send 3 packets (i.e., execute steps S2 through S4 3 times) is 3* (1,000 + 2,999 + 1,166) = 15,495 usec.
S6. Wait for an acknowledgement. While this does not require any processing time, it is necessary to calculate the amount of time that will be spent waiting for a response from the receiving iqdftp. This will be the time it takes for the receiving machine to transfer the third packet from NIC2 to its iqdftp (step R2, determined to be 2,999 usec below), validate all 3 packets (step R4, 6,000 usec), and transfer the ACK to NIC2 (step R5, 1,529 usec). It will also include the 10 usec startup delay of step R6. The total idle time in step S6 is then 2,999 + 6,000 + 1,529 + 10 = 10,539 usec.
S7. NIC1 receives the ACK. 32 usec (same as qdftp step 12).
S8. NIC1 transfers the ACK to iqdftp. 1,529 usec (see qdftp step 13).
S9. Repeat steps S2 through S8 4 more times (15 packets per read). Since step S2 can begin immediately after S8 is completed, the total time to execute steps S2 through S8 5 times is (15,495 + 10,535 + 32 + 1,529) * 5 = 137,955 usec.
S10. Check if file sent. 10 usec.

Receiving side calculations:

R1. NIC2 receives a packet. 1,156 usec (see qdftp step 6).
R2. NIC2 transfers a packet to iqdftp. 2,999 usec (see qdftp step 7).
R3. Repeat steps R1 and R2 2 more times. Step R2 executes as soon as R1 is done, but then R1 does not execute again until the sending iqdftp sends it more data. This means the sending iqdftp must complete steps S2 and S3 (and the 10 usec startup delay of S4). This takes a total of 1,000 + 2,999 + 10 = 4,009 usec. However, during 2,999 of these usecs step R2 is executing, so the receiving iqdftp is only idling for 1,010 usec in one loop. The time to execute steps R1 and R2 3 times is then (1,156 + 1,515) * 3 + 1,010 * 2 (no idle time occurs after the 3rd packet is received), or 10,033 usec.
R4. Iqdftp validates the received packets. This requires back-to-back validation of 3 packets, which takes 3 * 2,000 = 6,000 usec. Note that it technically

would have been possible to improve coprocessing by beginning this step during the 2,020 usec of idle time in step R3, but this was left out to reduce complexity.

R5. Transfer an ACK to NIC2. 1,529 usec (see qdftp step 10).

R6. Send ACK from NIC2 to NIC1. 44 usec (see qdftp step 11).

R7. Repeat steps R1 through R6 4 more times. As in step R3, it is necessary to calculate the idle time introduced by waiting for the sending iqdftp between the completion of step R6 and the beginning of step R1. This involves the sending iqdftp executing steps S8, S2, S3, and the first 10 usec of S4, for a total of 1,526 + 1,000 + 2,999 + 10 = 5,535 usec. Total time to execute steps R1 through R6 5 times is then (10,033 + 6,000 + 1,529 + 44) * 5 + 5,535 * 4 = 110,170 usec.

R8. Write the received 15 packets to disk. 52,220 usec (see step S1).

With these figures in hand, we can now calculate the total time to transfer a file. This can be done by calculating the time it takes the sending iqdftp to read the first 15 packets, send them, receive the corresponding 5 acknowledgements, and then must include any idle time before it can read the next packet. In terms of the steps above, it is the time in step S1 (time to read 15 packets), the total time we calculated in S9 (time to send 15 packets and receive acknowledgements), and also step S10 (end of file test). There is no idle time between step S10 and the next execution of S1. The total number of disk reads in 809,317/(1474 * 15) = 37 reads.

It is not necessary to include figures calculated on the receiving side because their impact on overall throughput are indirectly included in the idle times included in the S9 calculation. Note that step R8 (writing received data to disk) is not included in these number because the sending iqdftp is never waiting for this step to continue its processing – it is always doing something else at the same time. Technically, however, we should include the time for the final disk write in our calculations because during the final write the sending iqdftp will already be done and "total" completion waiting for the write to finish. Total time for the transfer is then (52,220 + 137,940) * 37 + 52,220 = about 7 seconds. Note that we could also have calculated this time by looking at just the receiving machine numbers, since again they already reflect idle time caused by waiting for the sending machine.

(iii) Are there any disadvantages to your proposed changes? *[4 pts]*

In general, disadvantages to recommendations for improvements suggested in many of the student homeworks tended to be in the area of "network friendliness". Although performance is a key consideration, the impact that the application has on the network as a whole must be considered, as well. In this light, sending a single burst of 809,317 bytes across an ethernet at one time (even if it were technically possible) is probably not a good idea. In general, ethernet network administrators try to keep network utilization around 20% to reduce the impact of collisions.

A related problem has to do with errors. If large packets are being sent, or if the number of acknowledgements are reduced, then the impact of errors and noisy lines are increased.

An obvious disadvantage of this new iqdftp is that it is much more complex which means that it is more difficult to design, implement, and debug. Of course, that is only a disadvantage if <u>you</u> have to do the design, implementation, and debugging!

While the new iqdftp is not a "network hog", there are still several aspects that could be improved. Its biggest weakness is that it doesn't take into account that the sending and receiving machine might be significantly different speeds. For instance, if the sending machine is a high speed machine with an "intelligent" ethernet card and sophisticated disk subsystem, the time it takes to read and validate data and transfer data back and forth with the NIC would be dramatically reduced. This could mean that it would be sending the next packet before the receiving machine had transferred its old packet off of its NIC. This may result in the receiving machine "missing" the next packet, making it necessary to retransmit the packet. Some form of pacing or up-front arbitration might be a way to improve on this.

While not a "problem", per se, there is also room for improvement in terms of reducing idle time and optimizing the number of packets read every disk access and sent before receiving an acknowledgement. Reading 15 packets per disk access and sending 3 packets per acknowledgement were arbitrary figures that may be optimizable or even dynamically determinable (i.e., different for each file/network).

Size of file X that was sent: 809,317 Bytes

Size of the file packet qdftp reads from disk: 1474 Bytes

Time to read and write data from and to the disk: 8,000 μsec + 2 μsec/Byte
(1 μsec = 1 x 10^{-6} sec, which = 0.000001 sec)

Time to transfer data between qdftp and the NIC card: 1,500 μsec + 1.0 μsec /Byte

Overhead data added to each packet by the qdftp software protocol: 25 Bytes

Speed at which the NIC transmits & receives over the Ethernet: 10 μsec startup delay, then 10,485,760 bits/second

Overhead added to each packet by the NIC card: 16 Bytes
[Note: this overhead is added after packet is received from qdftp and before it is sent, and then is removed by the NIC card on the other side before giving the packet to the receiving qdftp.]

Time for qdftp to build headers and analyze packet to send: 1,000 μsec

Time for qdftp to validate a received packet: 2,000 μsec

Size of an acknowledgement packet: 1 Byte

Time for qdftp to determine whether it has sent the entire file: 2 μsec

Table 1. Speed and overhead data involved in qdftp

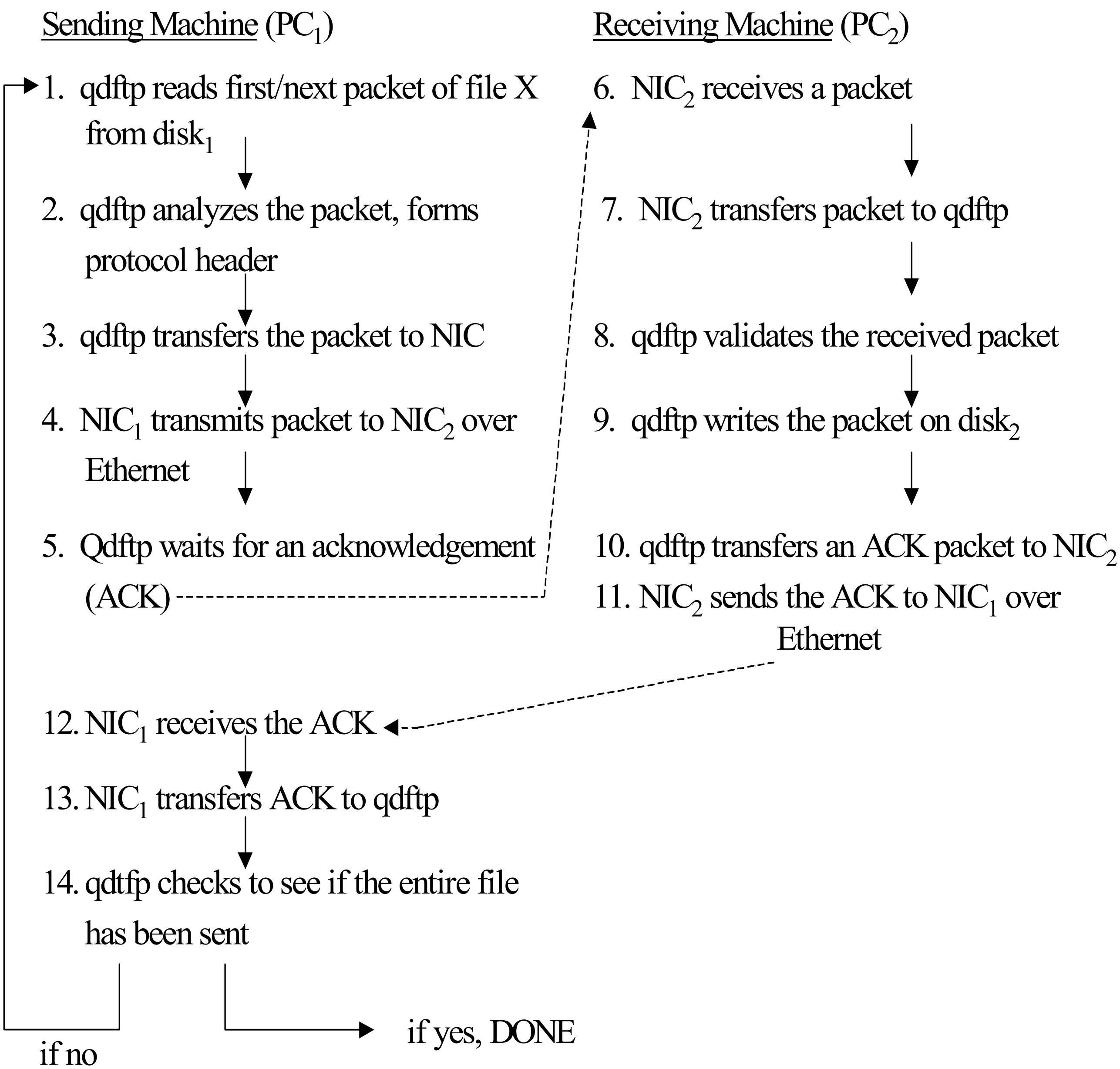

Figure 2. qdftp send and receive procedures