WHITE PAPER

THIN-CLIENT VS FAT-CLIENT COMPUTING

Thin-Client Versus Fat-Client Computing – McKenna's Four Laws

Definitions

Fat-client computing refers to a multi-tier client server paradigm where (in the simplest model) the client part of the application (i.e., those programs used by the end-user) execute on the desktop PC and the server part of the application (i.e., those programs used to drive the relational database) reside on a single server together with the application code.

Thin-client computing refers to multi-tier client server paradigm where the client (end-user) programs display in a browser (such as Internet Explorer or Netscape) but the execution of that user code takes place on a central web server, not at the desktop PC.

McKenna's Four Laws

1. For identical applications, the move from a Fat-client model to a Thin-client model does not reduce the need for computing resources; instead it moves the need from the desktop PC(s) to the Web server.

2. For identical applications, the move from a Fat-client model to a Thin-client model does not reduce the need for bandwidth; instead, it moves the need from the LAN/WAN infrastructure to the Intranet infrastructure.

3. For identical applications, the processing power required to service x number of users will be identical for Fat and Thin client models.

4. For identical applications, the move from a Fat-client model to a Thin-client model will not reduce the costs of the application and it may in fact increase the cost of the application.

1. No Reduction in the Need for Computing Resources

A Fat-client Application utilizes existing computing resources such as desktop PCs and LAN (network) servers. In a typical Fat-client Application the majority of the processing workload is handled by the desktop PCs.

For example, in a 500 user Fat-client Application roughly ninety-percent of all daily processing (execution of coded instructions) will take place on the desktop PCs as users asynchronously run the application at 500 locations. The remaining ten-percent is handled by the application/database server.

Rolling out a Fat-client Application usually involves utilizing existing capacity on already installed desktop PCs. This model allows for a relatively small application/database server (there may be two servers, one for the application and one for the database). It also allows for a significant number of new users to be added to the application before a server upgrade is required.

When an application is moved from a Fat to a Thin model; a Thin-client Application, the need for processing power doesn't go away. The same amount of processing capability must be provided. Because we are no longer utilizing the power of the installed desktop PCs it is necessary to install a server(s) large enough to replicate the amount of processing that would have been required in a Fat-client model. The new server must be powerful enough to do the processing previously done by a collection of desktop PCs.

In summary, Fat-client computing requires a relatively low power server because the bulk of the processing is done on the desktop PCs. By contrast, Thin-client computing requires a high power server because it has to do all of the work previously done by desktop PCs.

2. No Reduction in the Need for Bandwidth

It is a popular misconception that the move to Thin-client Computing (TCC) reduces the need for expensive communications infrastructure.

Anyone who doubts the need of TCC for expensive infrastructure should try browsing the web during peak times or talk to any Internet service provider about the cost of the infrastructure required to support a host site.

Effective TCC requires a significant investment in infrastructure in the form of servers, routers and high-speed connections to the Internet/Intranet not to mention the high cost of the backbone in the case of a private Intranet.

3. The Processing Power Involved Is Identical

For identical applications, if it takes x units of processing power to service a single user on a Fat-client Application it will take the same amount of computing power to service that same user on a Thin-client Application.

In the case of a Fat-client Application the bulk of the processing is done by the desktop PCs. In the case of the Thin-client Application the bulk of the processing is done by the web server.

4. The Move from a Fat-Client Application to a Thin-Client Application Will Usually Increase the Cost of the Application

When we move a Fat-client Application from 500 workstations we don't strip out RAM and half a CPU and send it back to the vendor for a refund. So, there are no immediate savings in computer hardware. However, the Thin-client model requires a significant new investment to implement and this cost is on top of all existing costs.

If the Fat-client Application involved loading the application on each and every workstation (a bad practice) we may reduce our workstation maintenance costs because the move to a Thin-client Application will reduce the amount of software that has to be maintained on each workstation. However, if the Fat-client Application did not require the loading of its code on each and every workstation but instead loaded its code on distributed application servers (the preferred model for Fat-client Applications) then we will realize very little savings with our move to a Thin-client Application.

In reality, the moving an application from a fat-client model to a Thin-client model saves very little on workstation maintenance because the major cost is in maintaining the network, Microsoft Windows, Microsoft Office and E-Mail systems such as Outlook. We still therefore need the same number of human resources to effectively manage our pool of desktop PCs.

Why Does The Cost Of A Thin-Client Application Often Greatly Exceed The Cost Of An Equivalent Fat-Client Application?

Once we accept McKenna's four rules we begin to realize that the problem is in:

- Replicating the power of a desktop PC for each on-line user; and
- The incremental cost of each new user.

When we add users to a Fat-client Application we use exiting PCs to absorb 90% of the load and reduce the load on the application and database servers. When we add users to a Thin-client

Application we are required to provide new, additional capacity or suffer a degradation in service to all users.

In a Fat-client Application processing power is decentralized, the environment is not single-point sensitive. In a Thin-client Application processing power is centralized and we are single point sensitive. A Thin-client Application is therefore much more likely to exhibit a sudden and exponential drop in performance than is a Fat-client Application. The realities of queuing theory cannot be ignored. As the resource nears 80% capacity more and more computing resources must be devoted to load balancing and 'swapping' and less to the actual execution of user code. With a heavily centralized system model response times tend to increase in an exponential fashion once peak loading (say 80%) is reached.

In a Fat-client Application we can make do with a relatively small and unsophisticated server. In a Thin-client Application we must have a relatively large and complex server. This complex Thin-client environment often comprises multiple clustered processors to provide the power required to service a large number of on-line users. This clustered server environment is extremely expensive to install and maintain and requires significantly more environmental and human resources than the lesser powered and less complex server used to support a Fat-client Application.

Diagram 1

Users	Thin Processor	Fat Processor
100	100	10
200	100	10
300	100	10
400	100	10
500	100	10
600	100	10
700	100	10
800	100	10
900	100	10
1000	100	10
Totals	1000	100

In the above table we see a clear explanation of why large scale Thin-client implementations rarely 'save' money.

For every additional 100 users in a Thin-client Application we have to add 100 units of processing power to the central web server. This is because we are not utilizing the power of the desktop PC to absorb some of the need for additional processing capability.

For every 100 additional users in a Fat-client Application we need only add 10 units of processing power to our server because we are utilizing the existing power of the desktop PC to absorb 90% of the requirement.

In the table above, we have added 1,000 units of processing to support 1,000 users in our Thin-client processing model but were only required to add 100 units of processing (one tenth the cost) to support 1,000 users in a Fat-client processing model.

So, Why Do Organizations Move To Thin-Client Computing?

They move because they think they will:

1. Reduce the need for bandwidth and relieve the load on the WAN;

2. Reduce the maintenance costs of desktop PCs;

3. Have more control over the application;
4. Provide better (i.e., shorter) response times;
5. Make use of the latest hardware and software technologies;
6. Make the application easier to roll-out and update; and
7. Save money.

Let's see what I think…

1. Reduce the Need for Bandwidth and Relieve the Load on the WAN

The need for bandwidth on the WAN is reduced (though we shouldn't rely on it because it is a universal rule in computing that usage will increase to the available capacity) BUT a new requirement for equivalent or greater bandwidth is generated for the Intranet infrastructure. Conclusion – no savings.

2. Reduce the Maintenance Costs of Desktop PCs

The maintenance cost for existing desktop PC's may be reduced if the previous application (because of poor design) required all executables to be loaded on the desktop PC. There will be little saving if the previous applications utilized existing LAN servers to store application code. Because the main headaches of network managers are still present (i.e., Microsoft Windows, Office, Outlook and networking software) an organization is unlikely to reduce the number of staff assigned to desktop PC maintenance. Conclusion – no savings.

3. Have More Control over the Application

The manager of a Thin-client processing facility may indeed have more control over the application BUT he/she now has a much larger and significantly more complex server complex to worry about and maintain. Conclusion – win some, lose some.

4. Provide Better (I.E., Shorter) Response Times

Remember that Thin-client computing requires significant resources; the need for computing power and bandwidth doesn't magically disappear just because we changed our processing paradigm from fat to thin. Because it is single point sensitive and because the incremental cost of a new user to a centralized Thin-client system is ten times the incremental cost of a new user to a Fat-client system, it is more prone to sudden and exponential increase in response times. Conclusion – we have just moved the problem, we have not solved it.

5. Make Use of the Latest Hardware and Software Technologies

Of course the programmers get to play with the very latest and most expensive hardware and software toys and their resumes look all the better for it. Conclusion – objective achieved.

6. Make the Application Easier To Roll-Out and Update

The application is easier to roll-out and update when web server based. Conclusion – objective achieved.

7. Save Money

Highly unlikely, especially when trying to replace an existing Fat-client Application. Realistically, any organization going the Thin-client route will actually incur significant new costs. Conclusion – saving money is a pipe dream.

The Real Question – Why Is Knowledgeone Corp Developing New-Technology Thin-Client Solutions?

Knowledgeone Corp is in the business of anticipating future trends and needs and then building products to service those same trends and needs. No matter what conclusions this paper arrives at, Knowledgeone Corp believes that Thin-client computing will be the dominant application-processing paradigm of the next five years. Knowledgeone Corp, unlike a Microsoft, is not big enough to affect the path of the IT industry. It therefore behooves us to provide the products our client's request.

Written by Frank McKenna, CEO Knowledgeone Corporation
2002