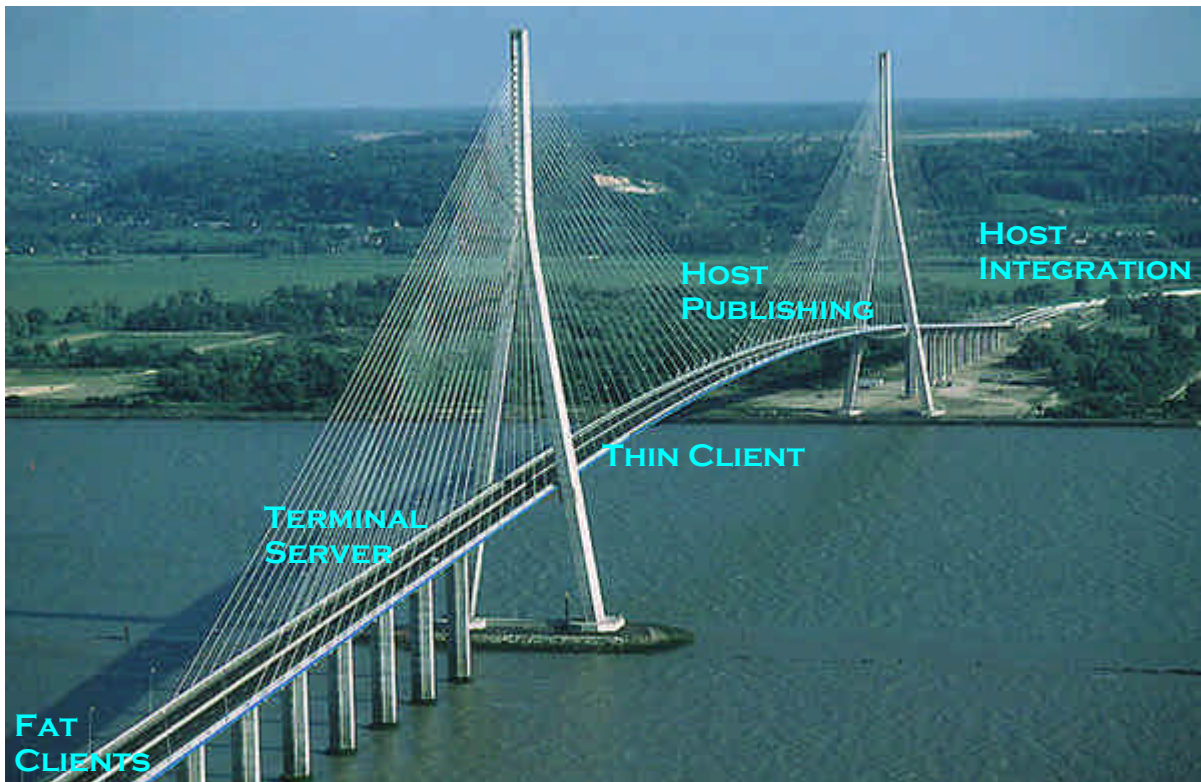


### ROI ESTIMATOR FOR HOST ACCESS MIGRATION OPTIONS

From Fat Clients to Host Integration  
taking in Thin Clients and Host Publishing



Created by: Anura ['SNA'] Gurugé

PUBLISHER I-BIGBLUE, AUTHOR, ANALYST AND RACONTEUR

# Table of Contents

- Hidden Costs Undermine ROI Estimates** ..... 2
- Consistent Approach Across All Options** ..... 3
- Quick Level Set About Fat Clients** ..... 5
- ROI Considerations for Fat Clients** ..... 7
  - Thumbnail sketch of fat clients ..... 7
  - Primary reasons for considering fat clients ..... 7
  - Key ROI accelerators for fat clients ..... 9
  - Key ROI delayers for fat clients ..... 9
  - Hidden costs in the case of fat clients ..... 10
  - Locking into the ROI of fat clients ..... 10
- ROI Considerations for Terminal Server-Based Thin Clients** ..... 12
  - Thumbnail sketch of terminal server schemes ..... 12
  - Primary reasons for considering terminal server..... 12
  - Key ROI accelerators for terminal server ..... 12
  - Key ROI delayers for terminal server ..... 13
  - Hidden costs in the case of terminal server ..... 13
  - Locking into the ROI of terminal server ..... 14
- ROI Considerations for Web-to-host Thin Clients** ..... 15
  - Thumbnail sketch of Web-to-host thin client ..... 15
  - Primary reasons for considering Web-to-host thin client ..... 15
  - Key ROI accelerators for Web-to-host thin client ..... 16
  - Key ROI delayers for Web-to-host thin client ..... 16
  - Hidden costs in the case of Web-to-host thin client ..... 17
  - Locking into the ROI of Web-to-host thin client ..... 17
- ROI Considerations for Host Publishing** ..... 18
  - Thumbnail sketch of host publishing ..... 18
  - Primary reasons for considering host publishing..... 18
  - Key ROI accelerators for host publishing ..... 19
  - Key ROI delayers for host publishing ..... 19
  - Hidden costs in the case of host publishing ..... 19
  - Locking into the ROI of host publishing ..... 20
- ROI Considerations for Host Integration** ..... 21
  - Thumbnail sketch of host integration ..... 22
  - Primary reasons for considering host integration..... 22
  - Key ROI accelerators for host integration ..... 22
  - Key ROI delayers for host integration ..... 22
  - Hidden costs in the case of host integration ..... 22
  - Locking into the ROI of host integration ..... 23
- About the Author** ..... 24

# ROI ESTIMATOR FOR HOST ACCESS MIGRATION OPTIONS

## HIDDEN COSTS UNDERMINE ROI ESTIMATES

Today, thanks to Web technology, there are more options for host access than has ever been available in the 35 year span of interactive host communications. All of the options are viable, proven and compelling. Nonetheless, they each have their own set of pros and cons, not to mention hidden costs, that significantly influence their potential return on investment (ROI). Backward compatibility can be an issue in many cases. Mission-critical client-side applications developed for fat clients [e.g. via an (E)HLLAPI interface] may not work with Web-to-host thin clients – even if the thin client and the fat client are from the very same vendor.

*Lack of backward compatibility between thin and fat clients even from the same vendor can be a big issue when it comes to ROI projections. A need for retraining, by itself, can push back your ROI estimates by a year or two!*

Significant differences in the user interface of a Web-to-host thin client from that of a fat client, even if they are from the same vendor, may necessitate retraining. Retraining is time consuming and costly. Even after retraining, a new user interface will inevitably impact productivity, reduce transaction volumes, increase error-rates and heighten the rate of calls to the help desk. Thus, an easy to overlook issue such as user interface incompatibility can make a huge dent in the projected ROI.

Web-to-host, Java- or ActiveX-based thin-client solutions offer some hard-to-ignore possibilities for reducing TCO. For a start, they tend to be less expensive than feature-packed fat clients. Not having to install and maintain them on each and every desktop can also result in some large cost savings as repeatedly talked about in [i-BigBlue](#). Then there are the huge cost reductions possible vis-à-vis remote access costs by using these thin clients across the Web to interact with host systems. However, much of these cost savings could be wiped out for years to come, if for example, you have to redo all of the macros, keyboard layouts and customizations required by the users.

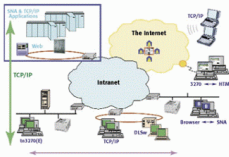
## A CONSISTENT APPROACH ACROSS ALL OPTIONS

The express goal of this document is to help you determine realistic ROIs for the various host access options factoring in the hidden costs. For each of the options you will be presented with:

1. Thumb nail sketch of how host access is achieved with this option
2. Primary reasons for considering this option
3. Key ROI accelerators: the well known and proven cost reduction possibilities associated with this host access option
4. Known ROI delayers: relatively easy to surmise factors that could increase TCO and delay ROI
5. Hidden costs: the unexpected costs that can wreak havoc to ROI projections
6. Ways to 'lock-in' on ROI expectations

This document is explicitly aimed at decision makers and influencers. It assumes that the reader already has a high-level technical appreciation of the underlying technology. This document is not meant to serve as a tutorial on contemporary host access technology. There is a full-page diagram on page 4, however, that sets out to summarize the key points associated with the different host access options.

### Integrating TCP/IP i-nets with IBM® Data Centers

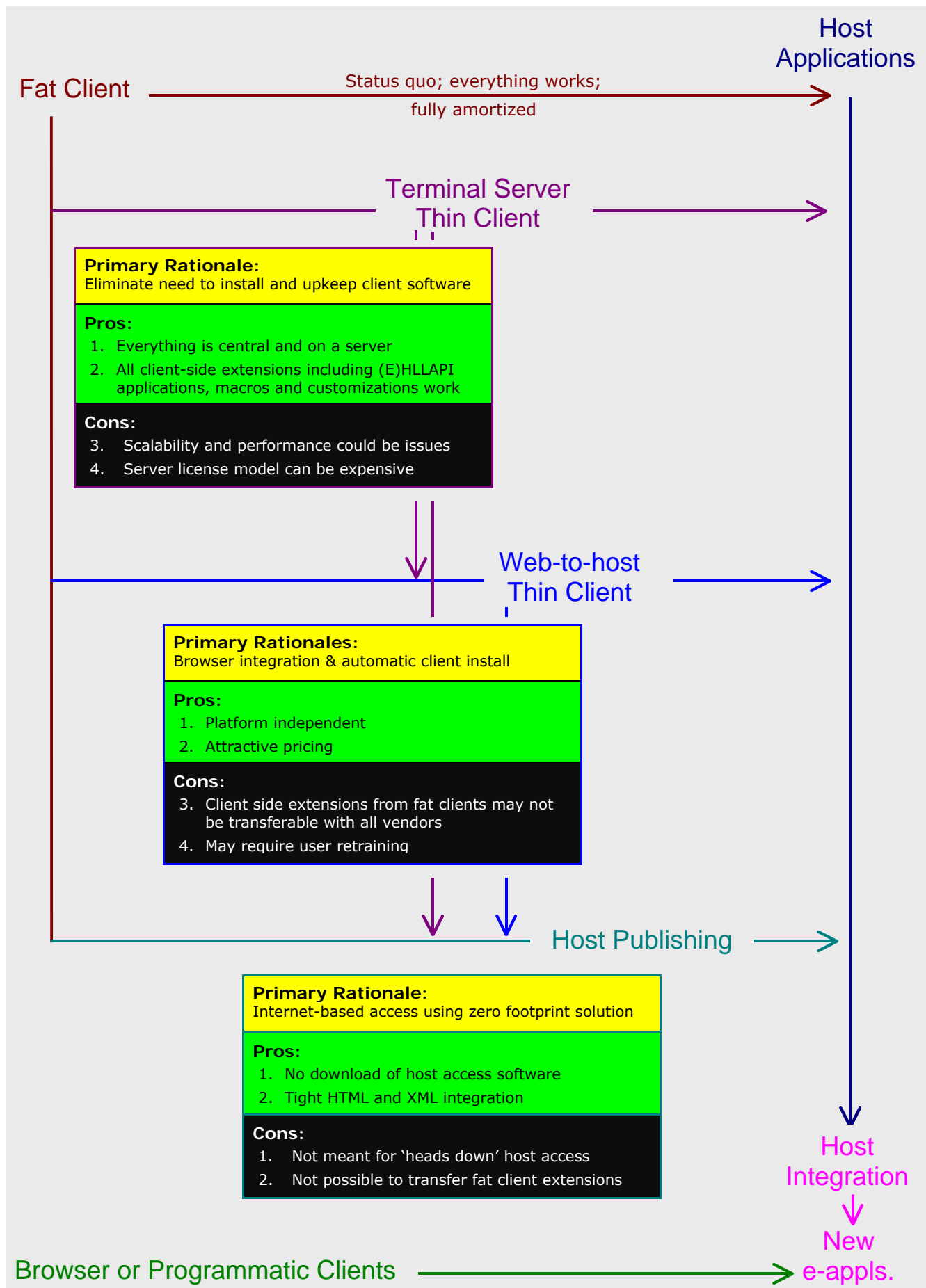


Anura Gurugé

Foreword by Don Listwin, Executive VP, Cisco Systems, Inc.

It also shows the various migration paths available from one option to another – including highlighting the fact that host integration is a means for developing new applications as opposed to being a host access methodology *per se*. In addition, the ROI discussions for each option will start with a brief description of the salient features of that host access scheme. If you do need in-depth guide to these host access schemes you could either look at the various documentation available from the key vendors [e.g. IBM, NetManage, Attachmate, WRQ etc.], or get your hands on my “*Integrating TCP/IP i-nets with IBM data Centers*” book. Refer to the last page of this document for information about this book and [i-BigBlue](#).

The host access options covered in this document include: fat clients, terminal server-based ‘thin clients’ [e.g. Citrix Metaframe or Microsoft NT Terminal Server], Web-to-host Java and ActiveX thin-clients, host publishing as well as host integration.



## A QUICK LEVEL SET ABOUT FAT CLIENTS

Traditional, feature-rich fat clients still dominate host access. There are multiple reasons for this pervasiveness, in addition to just plain inertia sustaining the status quo. No other host access option still comes close to offering the same depth and breath of emulation functionality, customization capabilities and networking flexibility available with fat clients.

The above claim is even true in the case of terminal server approaches which work by running a fat client on a server, as opposed to on individual clients. Moving the fat client to run on a server changes the networking alternatives available between the remote user and the host. Thus, if you need maximum functionality, heads-down data entry performance, customizability and networking flexibility [e.g. support for SNA, 'tn', IPX, 'split-stack'], a fat client may still be the best alternative for host access.

*Fat clients are still unsurpassed when it comes to emulation functionality, customizable capabilities, client-side application support, pedal-to-the-metal performance and networking flexibility.*

Another reason for the continued preponderance of fat clients has to do with the enormous size of the fat client installed base prior to the advent of the Web-to-host options. Fat clients were the accepted means for PC-to-host connectivity. They are reliable and resilient per the stringent requirements for mission-critical application access. They are also the basis for many heavily used client side applications. They are also the only host access option that supports SNA LU 6.2-based program-to-program [i.e. APPC] communications. Consequently, many companies have not seen any justifiable reason to get rid of fat clients.

The data throughput performance of a fat client is also unsurpassed. Heads down, high-volume data entry users will find other access schemes sluggish compared to what they are used to with fat clients. Constraining the data entry speeds of such users can impact transaction volumes and thus corporate bottom lines. Fat clients will continue to be the only option for heads down users as well as for the so called 'power users' who require all the bells and whistles when it comes to host access.

Fat clients also offer unparalleled host connectivity options – far greater than that available with thin clients. Thus they are the ideal client to have when an enterprise is in the process of migrating between networking infrastructures. The same fat client can work in

SNA, LAN, multiprotocol, IP and VPN scenarios. This is of considerable value and comfort to IT managers and network administrators. They can concentrate on making necessary changes to the networking fabric without having to continually worry whether the desktop clients will work with the new network configuration.

Fat clients have to be the starting point for any meaningful discussion about host access ROI. Fat clients are the incumbents. The other host access options are thus all alternatives to fat clients. Consequently, when it comes to host access migration, enterprises in essence have four choices:

1. Stay with the existing fat client installed base – and if not already using 'tn' start thinking about reconfiguring the clients to work in 'tn' for tight integration within emerging IP-centric networking schemes.
2. Upgrade to newer versions of the fat client [e.g. Windows 2000 or XT] to gain 32-bit support, even better performance and added resilience.
3. Supplement the fat client base [upgraded or otherwise] with other host access options such as thin client or host publishing.
4. Replace the entire fat client installed base with a new host access scheme.

*One of the few justifiable reasons for displacing a fat client installed base in favor of a new access scheme would be to gain Web browser integration in order to participate in intranet, extranet or corporate portal scenarios.*

To-date, the last of these options [i.e. total displacement] has been the one least employed. This makes sense. If the fat client installed base is doing its job, as it has for the last decade or more, why get rid of it, especially if it can work, flawlessly, within new IP infrastructures? Web browser integration, however, could be one of the few valid rationales for contemplating such a replacement. But there could be significant hidden-costs associated with such a move if you do not pick a backward compatible solution.

Some have considered getting rid of fat clients as a way to reduce host access related maintenance costs. This may not be as straightforward as it might appear – at first sight. There could be lots of hidden costs that would wipe out any savings over the first few years – not even counting the disruption of such a across the board migration, especially if you are dealing with tens of thousands of desktops. The ROI consideration of this and other migration scenarios are dealt with in the remainder of this document.

# ROI CONSIDERATIONS FOR FAT CLIENTS

## THUMBNAIL SKETCH OF FAT CLIENTS

Fat clients, also known as terminal emulators, have been the conventional means for realizing PC-to-host communications since the early 1980s. A fat client embodies all of the emulation and networking functionality required for host access – including in the case of SNA, a full SNA stack for direct mainframe or AS/400 connectivity without the need for an intermediary SNA gateway [e.g. Microsoft SNA Server]. They offer uncompromised terminal feature emulation [e.g. graphics and program symbol sets] and extensive customization capabilities including powerful and highly popular API's [e.g. HLLAPI] for desktop host applications.

*Fat clients are now the only host access option that supports LU 6.2-based program-to-program communications [e.g. CPI-C].*

Thanks to their strong SNA affinity they are now the only [not counting terminal server schemes] host access option that supports LU 6.2-based program-to-program communications [e.g. APPC, CPI-C]. Nonetheless, they are also premium, feature-rich [e.g. flexible printing options] 'tn' clients for contemporary TCP/IP-centric host access – with some now even following the lead set by Web-to-host thin clients by offering SSL security.

The most widely installed of the 40 [or more] fat client offerings available on the market include: IBM's PComm, Attachmate's EXTRA!, NetManage's RUMBA, WRQ's Reflection, Hummingbird's HostExplorer and Aviva's Aviva for Desktops.

## PRIMARY REASONS FOR CONSIDERING FAT CLIENTS

Fat clients, as discussed in the previous sections, are not a new option when it comes to host access. Instead they were the primary basis for host access, along with coax-attached 327x terminals, until around 1996. Fat clients thus need to be considered as the leaping off point for any other options. Consequently one does not have to justify fat clients *per se*. They are the incumbent, hopefully fully amortized, host access solution.

The only direct on-going cost related to fat clients should be the annual maintenance licensing -- assuming that this worthwhile 'insurance' has been maintained. Thus, ROI is typically not an issue in the conventional sense at this stage of the game. The fat client investment has already been made and it is now a 'sunk cost'.

Given their incumbency, there are in essence only two real choices that have to now be considered in the context of fat clients:

1. should the current installed base be upgraded to support Windows 2000, Windows NT etc?
2. should the need for additional clients be addressed with more fat clients or with newer options such as thin-clients?

*If you need additional host access seats should you continue with fat clients or should you consider other options?*

The upgrade issue, inevitably, comes up as a part of a much larger desktop standardization initiative – rather than as a host access specific necessity. Consequently, the ROI of such an upgrade cannot be centered around just the fat clients. They need to be upgraded because the enterprise is standardizing on a new desktop. So any ROI considerations have to be looked at from an overall holistic standpoint covering all other applications, increase in desktop stability etc. The key here from a fat client perspective, however, is to stay with the 'known'. Changing vendors could be detrimental to ROI.

Though fat client vendors promise conversion tools to migrate desktop applications, macros and customizations from competitive fat clients over to theirs this process is never as smooth as it is portrayed to be. Furthermore there could be user interface and operational changes that necessitate retraining. These conversion costs will directly erode ROI projections. Thus, in this instance, it would be best and safest to upgrade within the same brand – unless you can get an explicit 'price break' from the new vendor to defray all the costs associated with the conversion and retraining – including loss of productivity and associated 'lost opportunity costs' [e.g. higher error rates].

The same considerations apply when considering new options as a means of addressing the need for new 'seats'. Unless there is 100% backward compatibility with the fat client base, you will invariably incur additional – and often unexpected – conversion, training, support and help-desk costs. Backward compatibility covering all

facets of a client solution [e.g. user interface, macros, customized keyboard layouts, APIs], as mentioned at the start, is not a given – even between offerings from the same vendor. The only vendor that currently offers guaranteed backward compatibility between fat and thin clients, covering all aspects, from user interface to keyboard customizations, is NetManage -- with their RUMBA family.

## KEY ROI ACCELERATORS FOR FAT CLIENTS

Fat clients given their long-term incumbency are in a unique position when it comes to ROI considerations. They have most likely started generating positive returns a long time ago. If anything, by now, they are likely to have paid for themselves many times over – especially given their role in sustaining mission-critical operations. This would even be true in terms of justifying the ongoing maintenance costs. Given their role, a \$60/year per seat maintenance fee seems a bargain. Lets face it. You could be losing that per minute, if reliable communications to some of your mission-critical applications were disrupted due to client failures.

*Upgrading to the latest version of a fat client can improve productivity through gains in performance and resilience.*

Despite their payback in the past, there are still a few ways that you can squeeze even more return from your investment in fat clients. These include:

1. Exploiting the powerful 'tn' functionality available within fat clients to move away from costly SNA infrastructures, replete with 374x communications controllers and SNA Gateways, towards IP-centric networking – including the possibility of securely using the Internet as a means of very low cost bandwidth for remote host access.
2. Explore centralized, sever-oriented schemes, now supported by many fat client solutions, for centrally managing software updates, reconfigurations and customizations.
3. Upgrading to the latest version of the fat client – especially if such upgrades are covered by maintenance – to gain further user productivity increases via even greater performance [e.g. 32-bit] and resilience [e.g. Windows XT vs. Windows 3.1].

## KEY ROI DELAYERS FOR FAT CLIENTS

This issue, as with ROI accelerators, is somewhat mute in the case of fat clients since ROI is likely to already be positive. However, it should be noted that the following could negatively impact the fiscal advantages of a fat client installation:

*Maintaining fat client bases from different vendors is costly, non-optimum and complicates network management and support.*

1. Trying to maintain disparate populations of fat clients from different vendors, each with its own set of macros and customizations – either due to M&A activity or attempts in the past to switch from one host access vendor to another in order to get better products or service. If you do have such a splintered fat client base, this might be a time to cut your loses and standardize on the most popular of the in-house fat clients – assuming of course, that this client has good migration tools to permit the other fat client ‘enhancements’ to be cut-over.
2. Persevering with very old versions of the fat client [e.g. DOS or Windows 3.1].
3. Not using server-based tools to realize centralized administration and ‘provisioning’.
4. Not having maintenance on the fat client base that precludes access to key updates that could improve reliability and performance.

## **HIDDEN COSTS IN THE CASE OF FAT CLIENTS**

Given their extremely long history in the enterprise world there really are no hidden costs *per se* when it comes to fat clients. The dynamics of fat client-based host access is well known and understood. Using old versions of the fat client could, however, be construed as a hidden costs. Though the old versions continue to work, upgrading to the latest versions will boost user productivity via better performance and reliability.

## **LOCKING INTO THE ROI OF FAT CLIENTS**

In the case of fat clients this is invariably more of a question of maximizing the already positive ROI rather than trying to make sure that you do get a ROI down the road. As mentioned above, the things that can jeopardize this maximization of ROI is making the wrong decisions about how to handle the need for extra seats or the need for other options [e.g. for Web browser integration]. The key

here is ensuring backward compatibility. The cost implications of not having backward compatibility will be highlighted in the following sections – as they pertain to the fat client alternatives.

# ROI CONSIDERATIONS FOR TERMINAL SERVER-BASED THIN CLIENTS

## THUMBNAIL SKETCH OF TERMINAL SERVER SCHEMES

These are the Citrix MetaFrame or Microsoft Windows NT Terminal Server based approaches for providing client functionality by running client software on a centralized server – as opposed to on each and every client. The terminal server schemes are not host access specific and will work with most client software packages – though some host access vendors, such as NetManage, WRQ and Aviva, offer fat client variants optimized for terminal server deployment. Since it is in essence a fat client running on a server, as opposed to a desktop, backward compatibility *per se* should not be an issue – provided, of course, that you do not plan to switch fat client vendors.

*Terminal server schemes, such as Citrix MetaFrame or Microsoft Windows NT Terminal Server, are pre-Web methods to deliver thin-client benefits.*

The terminal server schemes were essentially a 'pre-Web' methodology to obviate the need to install and maintain client software on each and every desktop. Instead, the client software would be installed on central terminal servers – and then be accessed from client machines using a universal application-access 'thin client' resident on the remote desktops.

## PRIMARY REASONS FOR CONSIDERING TERMINAL SERVER-BASED HOST ACCESS

Eliminating the need to install and maintain client software is the overriding rationale for terminal server-based schemes.

## KEY ROI ACCELERATORS FOR TERMINAL SERVER-BASED HOST ACCESS

1. Installing and maintaining host access software on individual desktops can be costly – as much as \$150 per desktop per user [though the server-oriented centralized administration capabilities offered by the likes of NetManage's RUMBA 7.0 lessen this cost].

2. Ability to postpone costly desktop hardware and OS upgrades and continue using old, 'under-powered', limited memory client machines since the actual client software is installed and executed on a centralized server.
3. Support for multiple different types of client machines, i.e. DOS, OS/2, Windows 3.1, Windows 9x, Unix/Linux, Apple Macintosh etc., ensure that enterprises with different desktops [e.g. result of M&A] can postpone decisions about standardizing all the desktops.
4. Integrated security that obviates the need to implement supplementary security measures between the remote clients and the data center.

## **KEY ROI DELAYERS FOR TERMINAL SERVER-BASED HOST ACCESS**

1. Loss in productivity due to non-optimum performance given that there is now an additional layer of server-centric processing.
2. Higher overall licensing costs compared to desktop solutions – in particular Web-to-host thin clients which are now the strategic alternative to this pre-Web approach.

## **HIDDEN COSTS IN THE CASE OF TERMINAL SERVER-BASED HOST ACCESS**

1. Performance and scalability could lead to the need to constantly upgrade the servers or to add more servers to a growing 'server farm'.
2. A single server failures will impact all users attached to that server resulting in compounded loss of productivity.
3. Providing all of the extensions and customizations required by different groups or individual users will result in many versions of the same 'client software' [i.e. fat client] having to be maintained on the server.
4. The servers may require more nurturing and sustenance as user volumes grow than was apparent during the evaluation cycle.

*Scalability and performance continue can be major hidden costs with this approach that erode ROI possibilities.*

## **LOCKING INTO THE ROI OF TERMINAL SERVER-BASED HOST ACCESS**

Terminal server-based host access has been superseded by the newer Web-to-host thin client solutions – especially now that there are Java and ActiveX thin clients that are 100% backward compatible with their fat client brethren in all aspects, whether it be the user interface, macros, key layout or (E)HLLAPI application support. The Web-to-host thin clients offer the key benefits of the terminal server approach – i.e. elimination of the need to install client software on each desktop, platform independence and integrated security – without the performance and scalability penalties. Web-to-host solutions have thus become the strategic and preferred way to realize contemporary thin client host access.

## ROI CONSIDERATIONS FOR WEB-TO-HOST THIN CLIENTS

### THUMBNAIL SKETCH OF WEB-TO-HOST THIN CLIENTS

These are ActiveX- or Java-based terminal emulators. They are Web browser-invocable and moreover can run, tightly integrated, within a browser window. In addition to browser support, they, like terminal-server schemes, obviate the need for the client software to be installed and maintained on each and every desktop. However, they differ markedly from the terminal-server approach in that they do actually run on the remote client machines. This mitigates performance and scalability issues.

*Web-to-host thin clients offer tight Web browser integration as well as dynamic down-load from a Web server which eliminates the need for installation on individual desktops*

Web-to-host thin clients are initially installed on a standard Web server. Users invoke them via clicking on a hot-link or 'button' on a standard HTML Web page displayed within a browser. The browser passes this request to the Web server which then dynamically downloads the thin client to the user's machine. In most cases [e.g. IBM, WRQ, NetManage, Aviva etc.] the thin client will automatically be cached on the hard drive of the machine. This precludes the need for repeated downloads – unless, of course, a new version of the thin client is installed on the Web server. Some of the better known examples of Web-to-host thin clients include: IBM's Host On-Demand, SEAGULL's WinJa and JWalk, NetManage's RUMBA Web-to-Host, WRQ's Reflection for the Web and Jacada's Interface Server.

### PRIMARY REASONS FOR CONSIDERING WEB-TO-HOST THIN CLIENTS

1. Browser invocation and tight browser integration for intranet, b2b extranet, corporate portal and host access via Internet scenarios.
2. Eliminate the need to install and maintain host access client software on individual desktop.
3. Platform independence with the Java thin clients.
4. Integrated end-to-end SSL security with most of the popular solutions from the major vendors.

## KEY ROI ACCELERATORS FOR WEB-TO-HOST THIN CLIENTS

*Web-to-host thin clients are less expensive per seat than fat clients*

1. Eliminating the need to install and maintain host access software on individual desktops can result in significant cost savings as discussed in the terminal-sever section above.
2. Support for multiple different types of client machines ensure that enterprises with different desktops can postpone decisions about standardizing all the desktops.
3. Tight browser-integration permits immediate participation in new e-business initiatives such as CRM or supply chain management thus increasing corporate productivity through enhanced efficiencies.
4. Lower per seat (on current session) licensing than with fat clients.

## KEY ROI DELAYERS FOR WEB-TO-HOST THIN CLIENTS

1. Lack of backward compatibility with existing installed fat client base resulting in the need to convert host access macros, key board layouts, customization preferences and even (E)HLLAPI applications. This could add considerable cost, wiping out much of the anticipated cost savings especially if dealing with a large user base.

Just to get a measure of things, assume that for every 20 users there are specific macros as well as a keyboard layouts that need to be converted. Lets be optimistic and say that it will take 45 minutes to do both types of conversion and try them out. *If you have just 4,000 users that would mean 200 conversion efforts that would end up taking 150 hours – minimum!* You still haven't even touched the client applications – and those are likely to take even longer to convert.

2. Lack of stability and feature completeness [e.g. printing on Mac platforms], particularly from offerings from the 2<sup>nd</sup> and 3<sup>rd</sup> tier vendors, that leads to significant loss of productivity.

3. If the thin client is not cached, there will be a software down-load delay each time a user tries to access a host resulting in lost productivity.

## HIDDEN COSTS IN THE CASE OF WEB-TO-HOST THIN CLIENTS

*User interface incompatibility with the existing fat client base is a major hidden cost when considering thin client options.*

1. User interface incompatibility with existing fat client base would necessitate the need for retraining. This immediately hikes up the cost and there are further hidden costs such as loss of productivity, increase in support calls and higher error rates. Training is always expensive – and invariably ends up costing more than anticipated, even without factoring in all the associated costs such as business process disruption and the catering costs [e.g. donuts and coffee] to keep users 'motivated' and focused during training.
2. Sluggish performance with some Java clients could result in lost productivity.

## LOCKING INTO THE ROI OF WEB-TO-HOST THIN CLIENTS

The key here is to opt for a thin client that offers 100% backward compatibility in terms of user interface, desktop applications, macros and other customization. This, however, is not as easy as it sounds. IBM's Host On-Demand does offer a 'bridge' product to provide HLLAPI application conversion. Most others don't even offer this. The only vendor, at present, that sets out to provide total backward compatibility is NetManage.

# ROI CONSIDERATIONS FOR HOST PUBLISHING

## THUMBNAIL SKETCH OF HOST PUBLISHING

Host publishing is the ultimate thin-client solution when it comes to host access. With host publishing there is no host access related software, whatsoever, at the client. Host access is realized using just a standard browser at the client-end. Hence the term zero foot-print. The conventional host access processes, using 'tn' protocols, are done by a server resident component, e.g. IBM's Host Publisher v3.5. The server component then converts the host transactions and data into either HTML or XML. It thus Web enables host applications by providing them with an HTML(/XML)-based Web page interface.

*Host publishing is best suited for extending existing host applications to the Web for 'casual' usage by the general public via a GUI interface.*

Host publishing is a proven and simple way to introduce existing applications into e-business workflow. For a start you can immediately extend some of your 'legacy' applications, replete with new 'point-and-click' HTML GUIs, to authorized partners and selected customers – over the Web. It can also be very effectively used to extend a company's reach to include new users around the world, enhances competitive advantage and reduces call center costs by providing secure, corporate portal-based access to host applications.

Some of the better known host publishing products include: IBM's Host Publisher, SEAGULL's TigerWalk, Jacada's Interface Server, Attachmate's WebPublish 3.0, Hummingbird's eGateway and NetManage's OnWeb.

## PRIMARY REASONS FOR CONSIDERING HOST PUBLISHING

1. Quickly Web enable host applications so that they can be easily accessed over the Web by users at remote sites, partners and selected customers.
2. Enable host access using just standard Web browsers.
3. Platform independent host access.

4. 'On-the-fly' XML conversion [by some products] that facilitate integration of host applications with new e-business processes.

## KEY ROI ACCELERATORS FOR HOST PUBLISHING

1. Easily Web-enable host applications for home banking, personal travel reservation, online investing etc., extending company reach and increasing company competitiveness.
2. Gain a jump start on e-business by Web- and XML-enabling host applications.
3. Gain tangible supply chain management related cost savings by making key host applications securely accessible to selected suppliers, partners and distributors.
4. Eliminate the need for host access software on client machines.

## KEY ROI DELAYERS FOR HOST PUBLISHING

1. Deploying and activating the host publishing server component may not be as simple as expected.
2. Generating an acceptable Web page-based user interface may prove to be quite labor intensive unless there are incisive 'GUI-facilitating' tools such as the Microsoft FrontPage 'plug-in' in NetManage's OnWeb.
3. Users accustomed to fat clients will require extensive retraining and even then there will be a decline in productivity for quite a long time as users come to terms with a whole new paradigm in terms of host access.
4. Performance will be slower, due to the bi-directional HTML conversion, yet again leading to lost productivity.

*Host publishing is definitely not suited for data entry users or for most 'power users'*

## HIDDEN COSTS IN THE CASE OF HOST PUBLISHING

1. Will not support client side applications since this is a server side solution that bears little resemblance to terminal emulator architectures.

2. No backward compatibility for fat client or Web-to-host thin client macros, keyboard layouts or customization. Any required customizations will have to be redone.
3. Network or server instability could result in lost productivity.

## **LOCKING INTO THE ROI OF HOST PUBLISHING**

Host publishing is optimally suited for Web-enabling existing applications so that they can be available to a new audience – as opposed to in-house users accustomed to accessing those applications via fat clients. Host publishing is not suited for heads down users. Power users may also find that they do not offer the host access features they require [e.g. extended graphics]. Printing options are also limited. So this is an option for bringing in new users or for e-business integration, as opposed to an explicit migration path for fat client users.

## ROI CONSIDERATIONS FOR HOST INTEGRATION

### THUMBNAIL SKETCH OF HOST INTEGRATION

Though representing the next [and possibly the final] chapter when it come to host access as it relates to the pre-Web [a.k.a. 'legacy'] mission-critical applications, host integration is not a host access scheme. Instead it is an innovative scheme for reusing the proven and valuable transaction processing business logic contained in existing mission-critical applications when developing new e-business applications or even Web services. Thus, it is a really an application development methodology.

*Host integration is an application development methodology rather than a host access scheme.*

Host integration is typically a 3-step process involving: transaction capture, transaction definition and transaction integration. The first two steps are realized using 'drag-and-drop' software development methodology via a 'designer component' running on a workstation. A developer familiar with the host application containing the required transactions will invoke that application from the designer component – using standard 'tn' host access. The developer will then navigate through the application and select the necessary transaction in terms of their constituent input and output fields. The designer component records the host access, screen navigation and the input/output fields that relate to each transaction.

Once the transactions have been 'captured', the designer component in conjunction with the server resident host integration module will define the exact host access, screen navigation and I/O interactions required to reuse that transaction in terms of a JavaBean or Microsoft COM object. Then comes the actual transaction integration phase, when the same developer, or a different group of developers, take the previously defined transaction, in JavaBean or COM form, and include them within new applications. For the new applications to be able to use these 'legacy' transactions the original host applications need to be running – as before. Thus it is a run-time transaction execution model, where the new and the old applications run concurrently. So in this context, the host applications get invoked by new application, behind the scene, as opposed to by actual end users.

Some of the better known host publishing products include: IBM's Host Publisher, SEAGULL's Transidiom, Jacada's Integrator, WRQ's Verastream Host Integrator and NetManage's OnWeb.

## PRIMARY REASONS FOR CONSIDERING HOST INTEGRATION

1. Gainfully reuse the proven and valuable business logic contained within existing host applications.

## KEY ROI ACCELERATORS FOR HOST INTEGRATION

1. Slash development schedules by being able to reuse existing and highly proven business logic.
2. Reduce the time taken to test new applications since the old transactions being reused do not have to be rigorously tested since they are unchanged.
3. New applications are more reliable and resilient, thus reducing lost opportunity costs and loss of productivity, since they contain less brand new code.
4. Further maximize the investment in existing host applications.

*IBM's Host Publisher and NetManage's OnWeb are the only products that combine both host publishing and host integration – and in addition also supports XML.*

## KEY ROI DELAYERS FOR HOST INTEGRATION

1. Transaction capture and transaction definition process is too complex and convoluted thus slowing down and delaying the development process.
2. Product instability results in loss opportunity costs and lost productivity.
3. Scalability and sluggish performance of the server component results in lost productivity.

## HIDDEN COSTS IN THE CASE OF HOST INTEGRATION

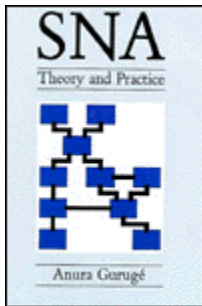
1. Most companies usually start with host publishing as a precursor to host integration – in terms of extending existing host

application functionality to the Web. However, only two major offerings, i.e. IBM's Host Publisher and NetManage's OnWeb, offer host publishing and host integration within the same product. Being able to do both functions, concurrently, with the same product reduces cost, simplifies overall operations and enables expertise and experience gained with host publishing to be leveraged when tackling host integration.

## **LOCKING INTO THE ROI OF HOST INTEGRATION**

Host integration, with the right product, can deliver very quick and dramatic ROI by significantly expediting new application development. However, to maximize this it is best to consider offerings such as IBM's Host Publisher and NetManage's OnWeb that offer both host publishing and host integration within a combined framework. That way you get the best of all worlds and the ability to reuse expertise and experience gained with host publishing.

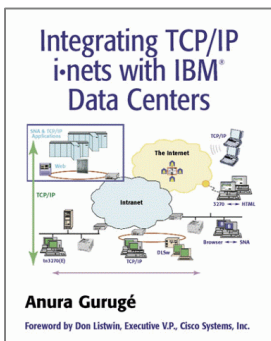
## ABOUT THE AUTHOR



Anura Gurugé is an Independent Technical Consultant who specializes in all aspects of contemporary IBM-related networking. He has first hand, in-depth experience in Web-to-host, SNA/APPN, Frame Relay, Token-Ring switching, ATM, System Management, and xDSL technologies. He was actively involved with the Token-Ring switching pioneer Nashoba Networks, which was acquired by Cisco Systems in 1996, and the ATM broadband access company Sonoma Systems which was acquired by Nortel in 2000.

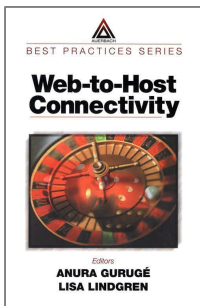


He is the author of *“Integrating TCP/IP i-nets with IBM Data Centers”* [pp 420, 1999], *“Reengineering IBM Networks”* [pp 600; 1996], the best selling *“SNA: Theory and Practice”* [pp 570; 1984]. He co-edited Auerbach’s handbooks on: *“Communications Systems Management”* and *“Web-to-Host Integration”*.



He is currently writing two books for Digital Press. One on *“Corporate Portals that support XML and Web Services”* and another on *“Web Services: A Guide for Decision Makers”*

He is the publisher of the monthly, 16-page *i-BigBlue Professionals’ Monthly* that has been available in various forms since 1995.



In addition, he has published over 280 articles. In a career spanning 26 years, he has held senior technical and marketing roles in IBM, ITT, Northern Telecom, Wang and BBN. He can be contacted at (603) 293-5855 or [anu@wownh.com](mailto:anu@wownh.com). [www.inet-guru.com](http://www.inet-guru.com) and [www.wownh.com](http://www.wownh.com).

