

# A Forms Perspective

Issue No. 3 - January 2002

## Solving the *'thin client'*—*'thick client'* dilemma

© 2002 Robert Barnett

The use of electronic forms is fast becoming one of the biggest fads in early 21st Century management. Coupled with the push for e-government, the desire for the 'paperless office' and speeding up service delivery, government and commercial organisations are looking to the World Wide Web's Internet and internal intranets for the simple solution. Yet, for all the enthusiasm, the technology is generally lagging far behind the imagination. Web technology is changing almost constantly, so we can expect massive improvements over the coming months and years. But for now, much of what management wants is no more than a grand pipe dream. Unfortunately, they often see how easy it is to access the Internet and don't realise that reading web pages is very different to filling out forms.

A further problem arises from the misguided ideas of many IT specialists who see electronic forms solely as a means of data capture, overlooking the need for business workflow (that's real business workflow, not just the movement of electronic data over a network), the need for keeping a permanent record of a transaction in its original format for legal purposes, etc. Data capture and electronic workflow are important, but good electronic forms systems require a much broader perspective.

### What this paper is about

The purpose of this short paper is to address some of the issues with a view to helping potential e-forms users make a more informed decision. One of the biggest dilemmas facing those who want to implement electronic forms is the decision about whether to use 'thin client' or 'thick client' (also referred to as 'fat client') technology. While 'thick client' software generally provides greater functionality and ease of use for the person filling out forms, 'thin client' has many advantages as well. It's not my purpose to promote one over the other, but to present what I hope is a balanced point of view, especially as I've found many people in the IT industry pushing 'thin client' as the ONLY solution. In other words, don't just assume that web browsers are the only way to use electronic forms.

## Some key definitions

Before looking at the issues, it's important to understand what we're talking about. The terms 'thin client' and 'thick client' are used in connection with the way a computer program's functionality is accessed. So here are some relevant definitions (some of the following information is based on material found on FOLDOC, the Free On-Line Dictionary of Computing).

### *'Client'*

This is a general term with various meanings. For the purpose of the present discussion, it can refer to a computer connected to a network that exchanges data with a server or it could refer to software that is used on such a computer. A more technical definition would be "a computer system or process that requests a service of another computer system or process (a 'server') using some kind of protocol and accepts the server's responses". A client is part of a client-server software architecture. For example, a workstation requesting the contents of a file from a file server is a client of the file server; an e-mail program retrieving e-mail from an e-mail server is a client of the server.

### *'Server'*

- A program that provides some service to other (client) programs. The connection between client and server is normally by means of message passing, often over a network, and uses some protocol to encode the client's requests and the server's responses.
- A computer that provides some service for other computers connected to it via a network. The most common example is a file server which has a local disk and services requests from remote clients to read and write files on that disk

### *'Thin Client'*

A simple program or hardware device that relies on having most or all of its functionality supplied by a network server. It is similar to a dumb terminal in that it gets all of its information from the network. For example, a simple HTML form filled out in a web browser is considered to be processed by a 'thin client' since much of the form's functionality is supplied by the server.

### *'Thick Client' (or 'Fat Client')*

A program that is stored locally on the user's computer rather than the server. For example, word processing software used to write letters and other documents generally resides on the user's computer rather than the server. Even when the software resides on the server it is actually on space allocated to the user and is, in reality, just an extension of the user's computer. The term can also be hardware related, referring to fast stand-alone PC's that have large amounts of memory and high volume hard drives that run programs locally rather than off the server.

## The Issues

This brings us to electronic forms and the way they are distributed to end users.

To start with, there is a fundamental issue that is misunderstood by many people. The next statement may seem childish to those experienced with computing, but it is one of the main issues I've had to face when providing advice to clients about electronic forms. **If you want to have intelligent forms, then the software to provide that intelligence MUST reside—even temporarily—on the form filler's computer.** Intelligence just doesn't pop out of thin air when you turn on the computer. Even if the software resides on a network server, it still has to be downloaded to the user's workstation and placed in the computer's RAM (random access memory) to be able to work. It is, in a practical sense, no different to having the software stored on the actual workstation since, even then, it has to be copied into RAM for it to be used. No matter what approach is used, there MUST be software to provide the forms intelligence.

So that leads to the big question. What type of software is needed to provide forms intelligence?

There are a number of possible approaches and we must understand them before deciding on 'thick' or 'thin' client. The following are some more common approaches, but are certainly not the only ones available. They are given as a guide to understanding the issue that the intelligence capability has to come from somewhere.

- A web browser capable of using JavaScript to add intelligence to the standard HTML. (**THIN CLIENT**)
- A Java-equipped web browser that downloads the intelligence applets (mini programs) each time the form is used. (NOTE: Java is not JavaScript.) Other languages besides Java could be used to achieve the same result. (**THIN CLIENT**)
- A web browser plug-in that loads a forms-capable program that can be read in the web browser. This may be a program that only works with a web browser or a program such as Adobe Acrobat Reader that will also work as a standalone program. (**THICK CLIENT**)
- A forms filler program residing on the user's computer. (**THICK CLIENT**)
- A programmed form (possibly created in a language such as Visual Basic) that has all the intelligence built-in. (**THICK CLIENT**)

Some people will argue with my third classification, but my argument is that if the program that provides the intelligence resides on the user's computer then it is 'thick client', even if the developer calls it a 'thin client plug-in'.

Some people will argue that even a web browser approach isn't really 'thin client' since the web browser provides the intelligence functionality and has to reside on the user's computer.

## **The cases for 'thick client'**

Using stand-alone forms software provides many advantages that are missing from the 'thin client' approach. This, of course, has to be weighed against any possible disadvantages of installation and maintenance.

### ***Functionality and flexibility***

To start with, there is usually much greater functionality and flexibility. The forms software can be developed to provide all the functionality the user needs without the limitations of third party software such as web browsers. While software such as Java and JavaScript can provide a lot of very useful forms functionality within the browser, it is generally much more limited than stand-alone software.

### ***Speed of opening***

Stand-alone software may make opening forms much faster, although that depends on the 'thin client' options. For example, an HTML form that uses only JavaScript to provide the intelligence will be fast to open and only take as long as it takes the HTML file to download. On the other hand, a Java-based form could take far longer to download since the Java 'classes' (mini programs) have to be downloaded each time the form is opened.

### ***Consistency***

Web browsers are notoriously inconsistent both in the way they display HTML on screen and in the

way they implement additional functionality such as Java and JavaScript. There are often major differences between Windows and Macintosh implementation as well as between different products (e.g. Netscape and Microsoft Internet Explorer) and especially different versions of the same product. Using a forms application (especially one that is cross-platform) overcomes this inconsistency.

### ***Saving form with data***

This is an important issue for some users, especially where they may need to refer to previously filled out forms on a regular basis or where there is a legal need to be able to show the data in the format in which it was originally entered. The representation of the data may be an important legal requirement.

With 'thin client' forms it is possible to store user data for later retrieval in the original format but this usually requires either specialised server/workflow software, or a lot of back-end programming.

### ***Large forms and off-line usage***

If forms are very long or the data the user is required to enter isn't readily available, this can cause annoying problems for the form filler. For users who are permanently on line to the Internet or their internal corporate network it may not be a big issue, but for public-use dial-up connections, the user may be forced to start again if the line drops out due to lack of use or other reasons. This can happen when the user has to leave the computer to find requested information elsewhere. One of the most successful long forms we have developed was for the Australian government's Department of Health and Aged Care. The 87 page application form collects a large amount of information from applicants for medical research grants. The forms can move from person to person in a research team and finally back to a central point for final checking before submission—and all this without any of the users having to be on-line after the initial download. This means that the team can compile its information at its convenience and only go on-line again when the final data is ready to be submitted. Because of the high functionality of the software, the form is self-checking so that form fillers know immediately if they have entered something that is invalid. This self-checking capability is also used to verify the form's content BEFORE submission. If errors are detected, the built-in intelligence prevents the form from being submitted.

### ***Installation***

In Australia, a number of government departments use a 'thick client' solution for hundreds of thousands of users with very few installation problems. In these cases, the users download the software from the government web site or get the software on CD-ROM and instal it themselves. Our experience shows that if the users are given simple and clear instructions and if the software is easy to instal, then even relatively inexperienced users have little, if any, problems. My argument on this has been that if members of the general public can instal forms software without any hassles then surely employees should be able to do it.

As I'll explain in the next section, this may be offset by the workstation configuration requirements making 'thin client' the only option.

A good example of the 'thick client' approach is NASA where both forms and software have to be downloaded and installed by end users. You can see how this works by visiting one of the NASA web sites listed below.

NASA Ames - <http://server-mpo.arc.nasa.gov/Services/NEFS/NEFSHome.tml>

NASA Marshall Space Flight Center (MSFC) - <http://starbase.msfc.nasa.gov/forms/welcome.html>

## **The cases for 'thin client'**

### ***Software installation***

I've discussed this issue with many computer people and the same major reason comes up every time: removing the need to install the application on every workstation. For some software this is indeed a valid reason as installation can be problematic. However, not all software installation is a problem as I'll explain below. 'Thick client' installation can be a problem in organisations that have poor networking support or where it is necessary to modify each client computer's configuration. The advantages can be offset by the need to instal suitable browser software. In discussing the issue with a number of customers, I've found that many (and sometimes all) of their users don't have web browsers capable of processing Java or JavaScript, or the latest releases of these applications. This is a particularly important issue for public-use forms where users will normally have a variety of browsers, browser versions and mixed platforms (i.e. Windows, Mac, etc.).

I've also come across a number of cases where the 'installation' argument is nothing more than an excuse to avoid travelling to remote locations to instal software. I'm not condemning the 'thin client' approach, just asking that people be honest when they give their reasons.

### ***Maintenance and support***

A second benefit is ease of maintenance, especially when there are new releases of the software and because there may be fewer demands on the hardware. With 'thin client', much of the maintenance is done on the server rather than the client machines. In some cases this may be offset by the need for the web browsers to be installed on client machines, and with some forms solutions, this means installing the most recent versions of the software. Ironically, this is often no different to installing forms 'filler' software and may actually take a lot longer due to the very large size of web browser installations compared to forms filler software.

There could also far less work for support staff who have to fix up the problems caused by end users messing around with installed software and changing configurations to what they perceive to be a better arrangement.

### ***Cross-platform use***

In some cases, 'thin client' provides a simple solution to cross-platform use. This is particularly important where the software used by an organisation can only run under a Windows operating system. This is particularly relevant with forms used by the general public. This isn't always a problem as shown in Australia. All the public-use forms we have developed for the Australian Government have been developed in Shana's Informed, which enables them to be opened on both Windows and Apple Macintosh platforms.

## **The case for a mixed environment**

One of the big problems in selecting software solutions is the desire of many organisations to use a single application for everything. From an expenditure perspective, this can seem like a good idea, but it can overlook the need for productivity, efficiency and validity of information.

A simple example of the need for multiple applications is the need for specialised word processing, graphic design and spreadsheet operations. Sure, you can create simple spreadsheets in the table facility of a program such as Microsoft Word. You can also use Word's graphic tool to draw simple images. But I don't think anyone would really try to do this for serious spreadsheet or graphics

purposes.

So what's the difference with forms?

The best solution—especially in a large organisation with thousands of forms—is often a mixed environment. As already explained, much will depend on the organisation's computing environment, the type and speed of the computer networks, the degree of decentralisation, the distribution of the organisation around the world and many other factors.

Many forms will no doubt work very well in a thin client environment while others may need to be thick client. Let me give you a typical example that comes from a large construction company implementing electronic forms. The following list illustrates the variety of forms they use.

- Many of their forms are simple one-page forms that can easily be filled out in a web browser.
- Some forms need to be in word processor format as they are essentially standard letters.
- Some forms need to be in spreadsheet format due to the very large amount of data that needs to be collected on major construction projects.
- Since some forms have to be completed in remote locations or in areas where computer access isn't feasible, there is a need for forms that can be filled out on paper, so some forms need to be in print only format. Many of these forms are in PDF format enabling users to print the forms in the office and take the paper copies with them on site..
- There is also a requirement for all forms to be printed out on paper from local PC's when remote telephone or network lines are out of action in an emergency.
- Some forms need to be completed by subcontractors. Given the complex nature of some of these forms, there is a need for a thick client solution for off-line completion.

This is not a complete list but it illustrates the need for a mixed software solution in a complex organisation.

### **For more information**

Contact Robert Barnett and Associates Pty Ltd at:

MAIL: PO Box 95, Belconnen ACT 2616, Australia

PHONE: (02) 6241 9022 or (INTERNATIONAL) + 61 2 6241 9022

FAX: (02) 6241 9023 or (INTERNATIONAL) + 61 2 6241 9023

E-MAIL: [info@rbainformationdesign.com.au](mailto:info@rbainformationdesign.com.au)