

User maintainable e-Learning content

ELearnExpo Paris, January 25-26 2004

A transcript of the paper presented by Tim Neill, Managing Director, TNA Ltd

(Tel: +44 (0) 1273 470003) email: tim@tnanet.com web: www.tnanet.com)

Overview

A practical to guide to planning and creating e-Learning projects which may be easily modified *internally*, without the further involvement of the content developer. The benefits can include:

- lower overall project cost
- internal control over much (but not all) of the content reduces maintenance cost
- rapid internal updates to content (faster response to change requests)
- no dependency on external 3rd parties for updating
- longer program shelf life

1. The Challenge

When a company employs a third party developer to create a bespoke e-learning program, the biggest risk they take is that the content will quickly become outdated and more money will have to be spent externally in repeatedly modifying it. Changes to your products, services, organisational structure and personnel as well as to your competitors and the marketplace all need to be reflected in the material used to train your staff. If an induction program starts with a video welcome message from your CEO and six months later he resigns, the program is obsolete. Who will update it?

The *distribution* of the latest content is simplified nowadays with the advent of intranet and web delivery, so you can at least be sure that all learners are using the same on-line version.

But the real challenge is this: how can a company with limited or no in-house development skills still keep the content itself current, using only standard Microsoft Office tools and *without* involving an external developer?

This presentation explores the background to this challenge and describes techniques which can make the dream – or at least large parts of it - a reality.

By 'content' we mean all the material that makes up an e-learning program, such as:

- screen text, data and URL web links
- illustrations and diagrams
- photographs of people, places and objects
- quiz questions and interactive exercises
- animated sequences
- audio clips
- video clips
- special purpose sequences (eg: QuickTimeVR interactive movies, 3D models)

Some types of content are simple to update; others may require specialist skills. So how do you tackle the problem?

2. The classical client-developer relationship

Figure 1 shows in overview the traditional situation in which a user organisation (the 'client') sub-contracts the creation of a custom e-learning program to a third party developer. The client describes its requirements, supplies content materials and arranges access to its subject matter experts. As development progresses, it reviews the program, requests changes and finally accepts and implements the finished program.

The developer uses his project experience and skills in applying its software tools to produce a completed program.

When the client needs to update the program for any reason it must call in the original developer (or another developer, if it has obtained the program source) who makes all the changes and delivers a new version.

This process involves continual expenditure, substantial delays in releasing updated versions and a natural tendency to 'group' numerous minor changes into single, large and infrequent updates, thus further extending the period during which learners are using outdated content.

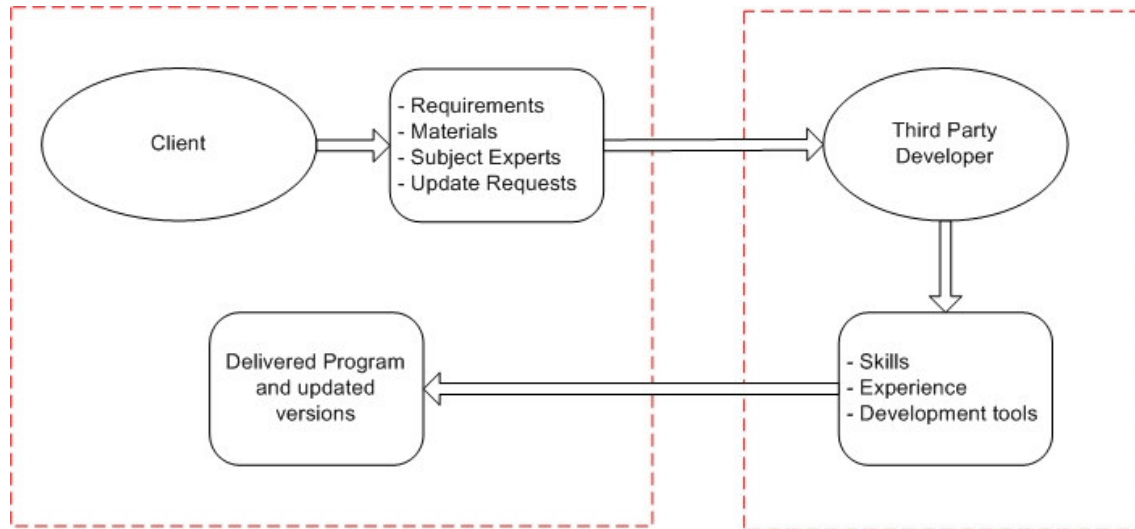


Figure 1: Program externally developed and maintained by a third party

Of course, it's obviously in the developer's best interest for their client *not* to be able to update the program content internally! After all, why would they willingly encourage their client's self-reliance rather than looking forward to years of lucrative revenue updating the program for them?

So, a program whose content cannot be updated in any way by the client remains a liability, an investment whose shelf-life may be very short.

3. What is 'user-maintainable content'?

By 'user-maintainable content' we mean all the material that a learner sees or hears as the program is used, rather than *the way in which the program operates*. So we do *not* include in this definition tasks such as adding menu options, altering the program structure or fundamental changes to the navigation. *As the program executes*, it fetches specified material from external files.

For example, the words which appear on menu panels, navigation buttons and in the screen text areas are read in from text files or from a database and dynamically inserted into variables ('holders') at the appropriate location.

Figure 2 below shows each party's responsibilities.

The benefits of user-maintainable content are considerable:

- Changes can be made rapidly to reflect your changing world
- Reduced cost of maintaining e-learning programs

- Programs have a longer shelf-life
- Excellent for efficiently producing multiple language versions without programming
- Reduced dependency on external developers (greater resilience in case they close down)

The content maintenance demands on the user organisation need not be great since:

- No programming skills are required
- Most changes will be quick and easy to make
- No expensive development tools are required
- Can use familiar MS Office tools

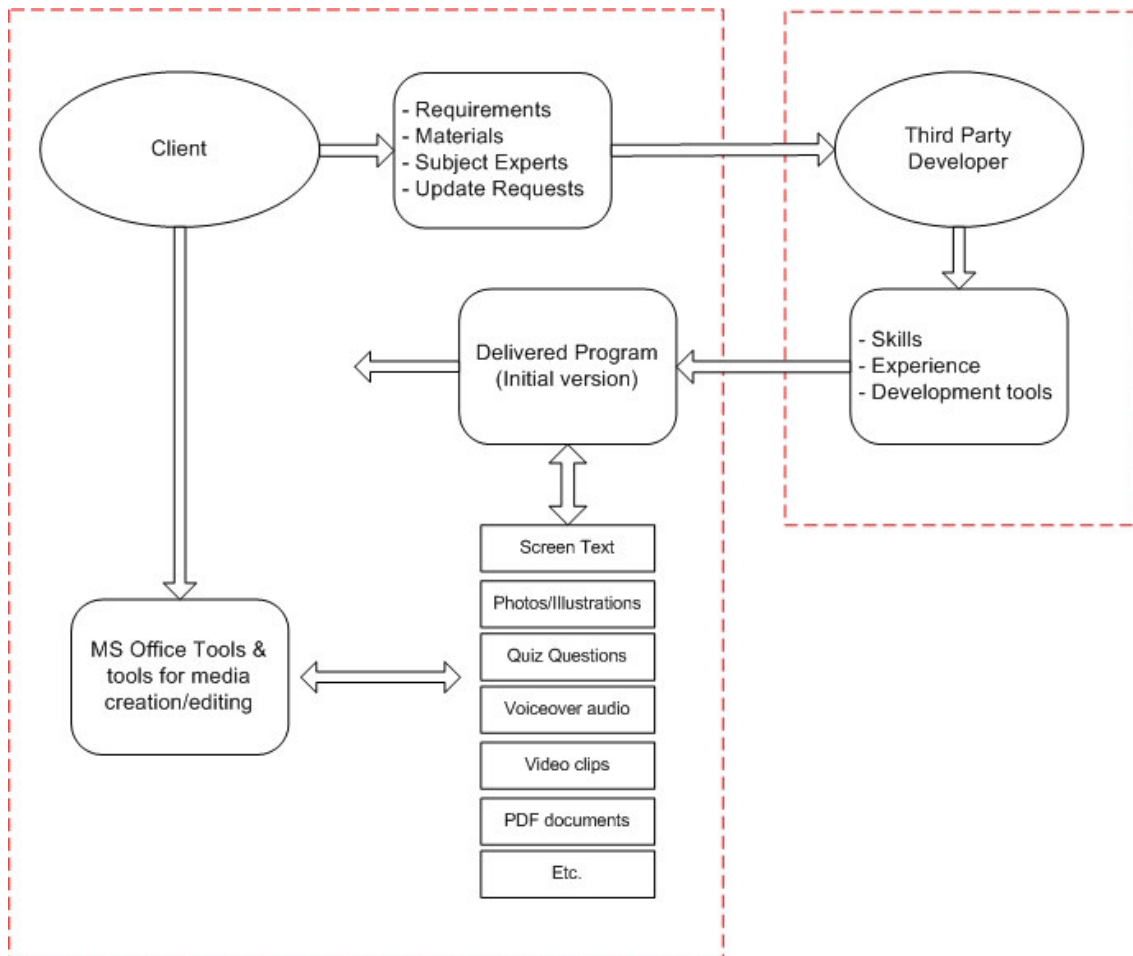


Figure 2: Program externally developed but with content maintained by the user

Referring to Figure 2, the key difference of user-maintainable content is that the material is, wherever possible, *held as individual files, separate from the program itself*, which renders them easily accessible for editing or replacement. We refer to this as 'external content'.

However, there are some drawbacks to this approach . . .

- Building e-learning that utilises external content (regardless of the quantity) demands additional planning, project management and continuing maintenance resources. It is the *user's* staff who will now be responsible for the program's continued quality and reliability.

- It will probably also involve additional external development costs, depending on the level of flexibility you require from the external content. A program whose content is 'bound' into the program (not accessible to the user) will usually be initially cheaper to develop.
- There is a greater risk of corrupting the program by the accidental deletion, corruption or omission of the external files. Great care must also be taken to define and observe the *format rules* that you and the developer have agreed for the content, for example ensuring that the image in a replacement graphic file is the correct pixel width and height, file format and file name. Or, the text appearing in title bars and menus which must not exceed a maximum length . . . too many characters and the word will overrun its allotted area on the screen. These types of rules *can* be checked automatically by the program but this all adds to the cost and complexity of the initial development.

4. Analysing your program content

The key to a successful project lies in finding the balance between the available budget and those areas of content which will need to be updated internally.

If you are too ambitious about the scope of what you want to be able to change, the development time and cost could rise substantially to pay for features that you may never need.

Conversely, if you *don't* plan for sufficient 'updateability', you may have to go back to the developers for expensive programming changes.

Example

An organisation commissioned a 60 minute e-learning induction program which introduced staff to their company. After analysis, they decided that they wanted to be able to modify all the on-screen text, all voice-over narration, the introductory video interview and the company's organisation chart.

- *Screen text was held in text (.txt) files, one for each screen, each 'bullet point' on a different line. Maximum text allowed per 'bullet point' was 100 characters.*
- *Voiceover was held as separate '.wav' files, named to match the screen and bullet numbers. Eg: C2T4B2.wav = the narration file for Chapter 2, Topic 4, Bullet Point 2.*
- *Welcome video was held as an '.mpg' file, 320 x 240 pixels in size.*
- *The organisation chart was held as an Adobe Acrobat .PDF file whose contents were displayed within the program.*
- *No other content was editable by the user.*

Other considerations

There are many other issues to consider when planning the project such as:

- Who will be maintaining the content and what skills will they need? (eg: scanning & editing images, voiceover recording, video shoots and editing, text entry, managing translations, etc.)
- Which *types* of content will need to be updated and how often?
- What are the *real* costs of sub-contracting this work versus producing it all in-house?

E-learning developers are already experts at all these tasks. Only if you are confident that your team can handle them to an acceptable standard should you take them on.

5. Notes on maintaining *basic* content in-house

The following types of content may be very easily maintained in-house:

- **Screen text, data and URL links**

In theory, *all* the text and data that a learner sees on screen could be held in external text files (or a database, discussed later) and edited using a utility such Microsoft Notepad or Word. These files should have been clearly labelled with comments by the developer to assist the editing process.

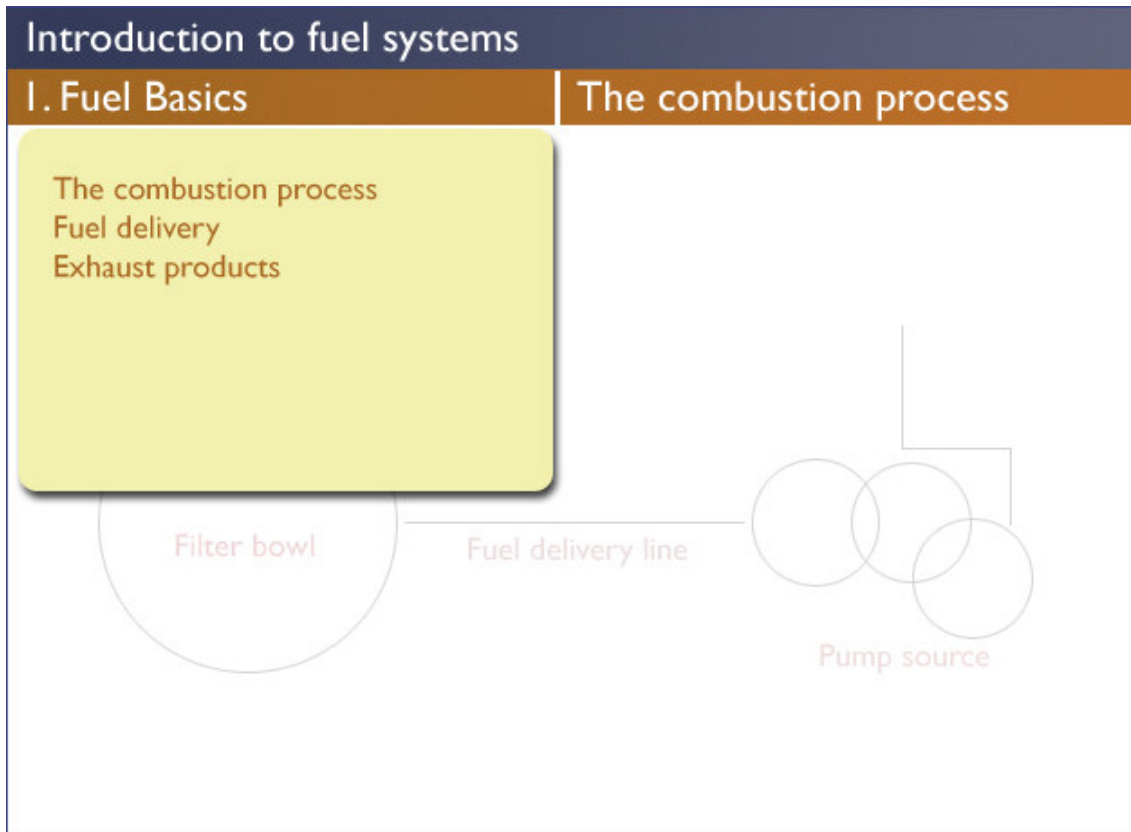
Consider the following example.

An e-learning program has four major chapters, each with several topics. The program's title and the names of the currently selected chapter and topic appear at the top of the screen.

A text file has been created by the developer called 'titles.txt' and whenever the program runs, it reads this file, fetches the titles and places them at their correct places on the screen. If the learner clicks on a chapter name, a 'drop-down' menu appears listing all the available topics in that chapter. The lines in the file 'titles.txt' might be structured like this (note the important '#' and ';' separators that the program uses to sort out the text):

```
Program title#Introduction to fuel systems#  
C1#1. Fuel Basics#The combustion process,Fuel delivery,Exhaust products#  
C2#2. Petroleum fuels#How are they produced?,Leaded fuel, Unleaded fuel#  
C3#3. Diesel fuels#How are they produced?,Applications,Advantages,Disadvantages#  
C4#4. Fuel pumps#Why are they needed?,Types of pump,The pump in action#
```

When the program runs, the title, chapter, topic names and menu panel text look like this:



So, a simple change to a text file would immediately modify the on-screen text although the text colour, font, size and style are pre-set within the program and cannot be altered. The same technique is used for storing the bulk screen text appearing in the actual learning content.

As a superior alternative to plain .txt files, .xml files allow an information 'structure' to be built which gives greater control over the way information appears. For example, *dynamically generated menus* can be driven from the text in an .xml file ... make a simple text change to a few lines and the program instantly displays *six* main menu buttons, not five. If the menu button graphics have been programmed in Macromedia Flash, the editable .xml file contents can even alter the appearance of the interface. Now *that's* user control!

- **Illustrations and diagrams**

Images may be held as external files in a format such as .BMP or .JPG. The program will be looking for a file of a specific name and extension; if it doesn't find it (because it has been wrongly named or is of the wrong type) nothing will appear. These images can be created using a program such as Paint Shop Pro or Photoshop. All the rules for their editing and creation should be agreed and documented in advance with the developer. If the developer has provided the *source* Photoshop files, this will make it far easier for your own staff to edit existing graphics.

- **Other linked files**

You may have other types of external file (eg: Acrobat .PDF, Excel or PowerPoint) whose contents are read by the program and used or presented in some way on screen. You can edit these files with Microsoft applications whilst keeping aware of the rules agreed for each.

- **Photographs of people, places and objects**

Stock library photographs and images captured with a digital camera will also be held in a format such as .JPG or .BMP. (The .BMP format is a 'lossless' format and is uncompressed. Therefore images can be very large and are not suitable for intranet or internet delivery. The .JPG format on the other hand allows you to vary the compression of the image – the greater the compression, the smaller the file but the poorer the quality. It is known as a 'lossy' format). Again, any image you replace must be the same pixel width height and width as the original and have the same name and file type.

- **Quiz questions**

Text for quiz questions and their feedback can be held in text files to allow easy editing. This needs great care in planning. If you alter the question text, which answer is now correct – 'B' rather than 'D'? The text file will have to hold details of this too.

- **Audio clips**

All voiceover narration and music should be held externally in a format such as .WAV or MP3.. You will need skill in using a sound editing program to modify or create new files. Great care must be taken to ensure that new/modified sound files are produced at the same volume level. Even then, recordings made in a different environment will attract attention. And an issue that is often overlooked: is the *original* actor still available for voiceover recording? If not, you may have to re-record the entire set of files to preserve consistency.

- **Video clips**

Video clips are always held externally in a format such as .MPG for CD-ROM delivery or Windows Media for streamed, on-line delivery. Video shooting and editing is a skilled task and is best left to a specialist company and you must keep to the rules on image width/height, the video compression codec and file name and type, as agreed with the developer.

If a video clip is accessed by the program using several questions selected by the learner (eg: an interview) then the video will be supported by a text file containing the text of the questions and the video frame numbers for the start and end of the corresponding answers. If you replace a video interview you will have to edit this text file too.

6. Notes on handling multiple language versions

Delivering training in *multiple* languages but using a *single, common* version of the program is one of the real benefits of external content.

The screen text and voiceover narration for each of your languages is held in separate directories, all with the same content but translated to the required language. Country-specific images can also be held in these directories, for example allowing the program to switch automatically to photographs of people, dress, buildings, etc. more relevant to the region.

The e-learning piece is initially created in one language, with a setting (possibly in a database, text file or a hidden windows setting) that specifies what language to use and what directory that language is in. Without any changes to the program, you simply add another language by creating relevant content in a new directory. As long as all the filenames match, it will all still run correctly. Make sure you have the original source documents (screen text and voiceover) to give to the translator.

The main issue with multiple languages is text length on screen. With direct translations, both German and Spanish in particular may cause problems on screen with the length of sentences.

If your e-learning piece relies more on voiceover, with short bullet points on screen, you probably won't experience difficulties. Languages that display from right to left – as long as this intelligence is built into the program initially - can easily be added later.

7. Dealing with more advanced content

The more complex the content, the greater skill you will need to have available in-house. The question is, how often will the item need to be changed and can you justify the required skill level?

Such content might include:

- **Animated sequences**

Ideally, animations should be created and held as external Macromedia *Flash* files. If you have the source code for these and the in-house skills to use *Flash*, then you would be able update the sequences or replace them with new ones.

- **Clickable images**

Images with embedded 'hot spots' such as a map of the world with user-selectable regional offices, will usually have been programmed rather than allowing you to modify the hot spots. But if you planned for this, and frequent changes are likely, then techniques such as 'image maps' in HTML or QTVR scenes (see below) can provide a solution allowing in-house modification.

- **Interactive exercises**

It is unlikely that you would wish to modify complex exercises (simulations, games, competence assessment, etc.) since this may require considerable programming skills and are unlikely to need modifying very often.

However, there is one format – QuickTimeVR – which holds 360 degree photo-imagery in an external file. These are sophisticated interactive movies, often with embedded 'hot spots' and linked scenes, which may be tightly bound into the way the program executes. They require specialist camera lenses and editing software.

QuickTimeVR can also provide an efficient way to create and edit simple .JPG images with hot spots, such as the map mentioned previously. The program will need to have been structured to deal with a *variable* number of hot spots and to fetch the relevant text or images (to be displayed when a hot spot is chosen) and display them in a set area. Not trivial but possible, if the additional effort and cost can be justified.

8. Content Databases

One of the most powerful ways of storing your external content is to use databases. The real power of this is in the *development* phase of your project rather than in the delivery.

Databases make it easier to manage, organise and build your content, rather than storing it in *text* files. Whereas both require the program to read in external information, it is in *referencing* the information that databases provide real advantages. You can simply have unique references to the pages, sentences or bullets that build up your content and pull them cleanly out of the database. Using the *text* file approach, the program has to sort, find and extract the information, based on embedded separation characters which you have to insert.

When it comes to modifying and correcting content at a later stage, a well organised database enables you jump straight in and quickly find the information you want to change. It is the elegant and preferred approach.

9. Delivery considerations

E-learning programs may be delivered in many ways, depending on the type of content and the speed and method of learner access.

▪ Program running direct from CDROM

If your e-learning piece is to run completely from CD-ROM, the main issue with delivery is this: once the CD-ROM's are sent out you can't update them without sending out a new CD-ROM. This can prove expensive if the program needs frequent updating. And how can you be sure all learners are using the latest version?

▪ Hard drive or hard drive/CD-ROM Hybrid

This approach is more challenging. For the greatest flexibility, an installation routine would automatically copy all files in certain directories and would not be restricted to specifying files and compressing them into one executable. The reason for this is that there may be files on the CD-ROM that are expanded upon at later stages (for example, a new language may have been added). So to improve development time, the installer or delivery method shouldn't specify individual files but *groups* of files.

▪ CDROM/On-line Hybrid

This is the most complex approach, but can be the best performing solution. All the *large* content (images, voiceover, video, etc.) can be held on CD-ROM or installed onto the users hard drive. But the program structure, screen text and other smaller files as well as all other information which may need updating in the future are held centrally on a server. This provides the best performance for the learner whilst giving you control over the volatile content.

10. Template-driven content – how far can you go?

The techniques described so far assume that the *layout* of each screen remains the same, just as it was programmed by the developer. The control you have in-house is over the words, images, audio and video which appear or play at their pre-destined position or moment.

But what if you want to be able to add a totally new screen to the program? Or to modify the colour, style or position of text? And to achieve this *without* programming skills? The answer is to create the e-learning program based on **editable templates** whose content is still derived from external files, as already described.

In its ultimate form, this approach would enable you to define the precise layout of every screen, with areas specified for video clips, images, animations and text and to save this as a new template. All template properties would be held in a database, with references to the external content to be used. The style, typeface, colour and size of text could also be set by you too, without any programming.

In fact, it is feasible to build a creation/editing tool that allows the entire contents of an e-learning program to be changed, including the background imagery, the interface and menu controls, positioning of items on screen, the displayed content and even the program's functionality.

The program would act as a 'shell' and would be completely controlled by the user via external content.

But this is moving towards creating a completely new development tool for in-house use which, in turn, uses another more complex development tool in the background.

It begs the question: would it not be easier (and cheaper) to simply use the *original* development tool such as Macromedia Authorware, and accept that your in-house staff will need to acquire and maintain the necessary skills with this commercially-supported system? In short, do you want to become a *developer*?

11. Summary

There is no limit to the scope and nature of the changes which you and your staff could make to an externally developed e-learning program. The key issues to face and answer are these:

- What material will need to be changed and how often?
- What skills do you have, or will need to bring in, to maintain the content?
- What expenditure can be justified, both initially and for future maintenance?

Tim Neill Associates (TNA) – eLearning Content Developers

TNA is a well established and respected developer of high quality, bespoke e-Learning programs, product simulations and interactive presentations.

Since 1988, TNA has produced dozens of effective and enjoyable staff training programs on topics from automotive sales techniques and asthma treatment to loan underwriting, communications equipment servicing and customer relations.

Our clients include Nokia, Phones 4u, B&Q, Nestlé, Emirates Airline, Vodafone, Orange (Geneva), Warner Music, Alfa Romeo, Land Rover and SITA Communications in London and New York.

TNA delivers the highest quality results, value for money and an outstanding 'can do' working relationship with its clients. www.tnanet.com carries summaries with screen shots from over 100 completed projects.