



EXTENDING THE LIFE OF SOFTWARE

Mark Claxton

TESSELLA SUPPORT SERVICES PLC

Issue V1.R2.M0

December 2001



EXTENDING THE LIFE OF SOFTWARE

Introduction

The purpose of this report is to focus on the benefits of updating outmoded user interfaces and data presentation methods with more current approaches.

By using examples we aim to show how organizations have achieved substantial productivity gains as well as enhanced customer and user satisfaction, through improving the interfaces to their existing code.

The Background

Many sites have a number of “legacy” systems that were written years ago. These systems often perform mission critical functions and are used on an occasional basis by a large user base. They took many years to evolve and their internal workings have been honed to perform the required tasks as reliably as possible.

These critical applications have often not kept pace with current computing practice and many suffer from poor or unfriendly user interfaces. In many cases they have a user interface that does not conform to any standard or that conforms to a standard that has since become obsolete. As a result, it is not uncommon for users to be faced with a mixture of interfaces with no common “look and feel”.

Visualization or results presentation is another area in which legacy systems may be showing their age. They were written to produce the results in the most effective and informative way using the tools available at the time of development. However software and hardware performance and functionality have been enhanced considerably over the last few years and it may be time to review the quality of the data presentation.

The use of outdated data presentation techniques and poor user interfaces can lead to a number of major problems including:

- poor productivity.
- increased error rates.
- delays caused by training new staff to use the legacy system.
- user dissatisfaction.

- maintenance and future enhancement problems.
- the need for “experts” to interpret results.
- inflexible tools that make it impossible for users to find vital data.
- presenting poor image to customers.

Recent Advances

Over the last few years, the Microsoft Windows environment has become a de-facto standard for user interfaces in the PC environment, Motif the standard for user interfaces in the Open Systems environment and SAA the standard in the IBM mainframe world. All of these standards were developed to provide the user with the easiest and most intuitive interface to the application. They all provide a common look and feel, ensuring that users on a particular platform are always presented with the same style of interface regardless of the application. This improves user satisfaction, can lead to improved productivity, reduced error rates and helps reduce training effort.

The improvements in data presentation and visualization make it possible for applications to present their results using ever more sophisticated techniques. The use of such visualization tools is becoming more common since they enable even novice users, to identify vital data more effectively. Data can be displayed in a number of different ways allowing users to identify trends or opportunities that may have gone unnoticed using less sophisticated techniques. Finally, they can produce a far more polished appearance to the end user and thereby present a better image to the customer.

How to Upgrade the Legacy System

There are a number of strategies a site can follow with regard to their mission critical legacy systems.

- Replace the old interface modules with up to date code using latest standards.
- Provide up to date front and back ends to the legacy system.
- Turn the legacy system into a compute server and give users interface clients.
- Undertake a complete rewrite and replace the legacy system.

There is no perfect single solution for all systems and any of the above approaches could be capable of providing the required result. However in most cases, the key deciding factor will be the cost of the upgrade against measurable improvements in productivity, accuracy, user satisfaction and customer

satisfaction.

Rewrite Interface Code

If the original legacy system was written with a robust and well defined interface between the underlying application and the user interface code, it is often a relatively simple matter to replace the old interface with an up to date implementation.

The figure 1 screen image represents typical old style input panels prior to the rewrite of the interface code. Figure 2 is a screenshot showing how the same data is entered using the new FoxPro based interface.

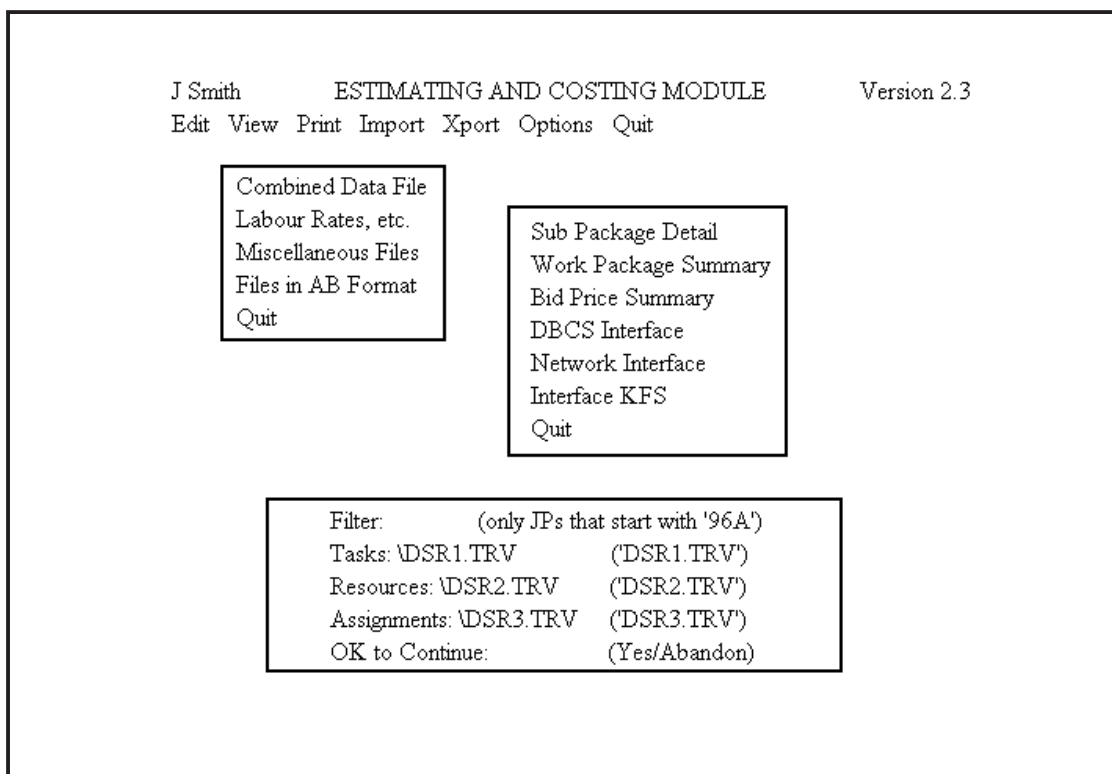


Figure 1: Example of an old style line model interface

This project was a system enhancement to a task costing system developed by Tessella. The application was written in Computer Associates Clipper to run under DOS. The underlying code was still critical to the organization, however the DOS based interface was out of step with the other Microsoft Windows based applications run on the site. The decision was made to convert the Clipper component to Microsoft FoxPro for Windows thereby bringing the application in step with the rest of the site. The key objective was to convert the line mode interface of the Clipper application into a Windows based interface.

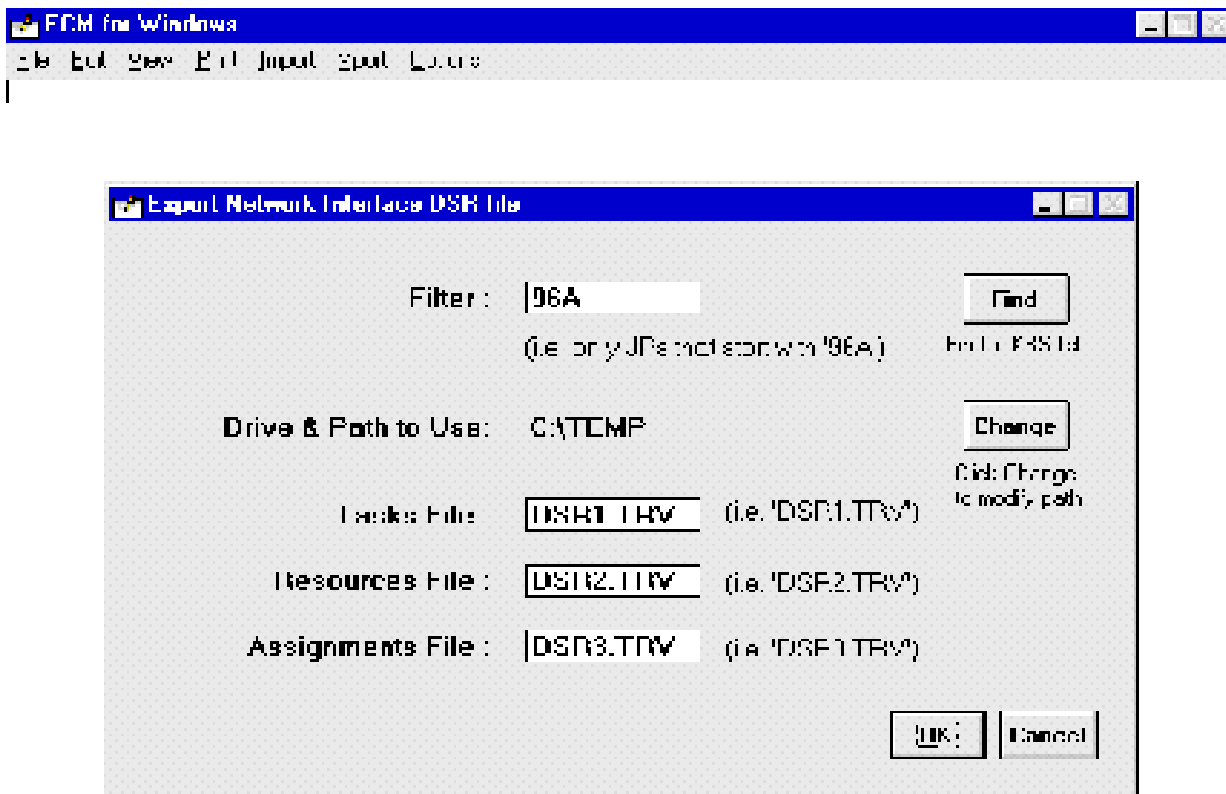


Figure 2: Using the FoxPro based interface

This upgrade took about 25% of the time to produce the original interface and has given the existing application a whole new lease of life. The customer reports benefits such as improved user satisfaction, decreased error rates and improved data entry rates. They also found that due to the shared interface standard, new users required very little training in operating the updated system.

Another example of this approach is the work Tessella did with a customer who manufactured machinery and also supplied the software used to control it. The software's original interface was text-based and ran under DOS – this looked dated, was not intuitive to use and was generally unattractive to their prospective customers. A new front end was developed using Microsoft Visual Basic to run under Windows NT. This gave their customers a more intuitive user interface (less training of operators required) and an online help system in case of problems. Improved security against operator misuse was also added.

Develop Front and Back Ends

In this scenario, it is not uncommon to be presented with systems that expect

data in the form of formatted input files and return results in the form of complex tables of numbers.

A cost effective solution, suitable for these applications, is to encase the original core system in a new interface module. In other words, you place a user friendly interface between the users and the formatted input file and pass the output results to a back end processor that interprets the data and repackages the information into a more understandable format. This solution ensures that the mission critical application is left untouched while substantially simplifying the way in which users provide data to the system.

Tessella has successfully undertaken a number of projects of this nature, with the following examples being typical.

```
'Test Site'
3
'AR-41' 0 3.29E+05
1.0
0.00E+00 0.00E+00 7.61E-02 'D'
3
1.000 4.090E-01

1.500 5.840E-01
2.000 1.840E-04
'KR-88' 0 5.10E+05
1.0
0.00E+00 0.00E+00 1.18E-01 'D'
7
0.100 2.050E-02
0.200 2.900E-01
0.500 7.190E-02
1.000 1.500E-01
1.500 1.670E-01
2.000 5.450E-01
4.000 9.180E-02
'RB-88' 1 5.10E+05
1.00E-03 1.00E-04 1.24E-02 'D'
1.0 'KR-88'
5
0.500 2.870E-02
1.000 1.140E-01
1.500 7.840E-02
2.000 1.650E-01
4.000 1.300E-02
50
9
1.000E+04 3.000E+04 4.000E+04 5.000E+04
6.000E+04
7.000E+04 8.000E+04 9.000E+04 1.000E+05
-1
4
```

Figure 3: The original input file

A customer had a sophisticated environmental simulation program that took physical data and control commands from a formatted input file. The program ran through the simulation and generated a series of tables which were then interpreted by the users. The simulation code took many months to develop and the customer had a high level of confidence in its accuracy and reliability. Unfortunately the results generated were frequently incorrect and this was found to be due to operator error in setting up the simulation. After analysing the situation it became clear that a more user friendly input interface was required that could be used to prompt for related data and that could check the input before passing it to the simulation code.

Figure 3 shows the format of the input file created by the user to feed into the FORTRAN simulation code which was run on a DEC VAX. After running benchmarks it became apparent that the code would run equally effectively on a high spec PC, so the simulation was ported from the VAX on to a PC and the FORTRAN code was given a

Other Information

Site Details:

Site Name: Effective Release Height (mm):

Distance Data:

Number of Distances:

	Distance (km)
1	10
2	20
3	40
4	50
5	60
6	70
7	80
8	90
9	100

Wind Data:

Uniform
 Site Specific Met. Data

Heightwise Length:

0.10m
 0.04m
 0.10m
 0.30m
 0.10m
 1.00m
 4.00m

Stability Category Classification Scheme:

Pasquill Husker Dyoung

Cancel
OK

Figure 4: The Visual Basic front end

Microsoft Visual Basic front end as in figure 4.

This upgrade proved very effective, as the number of incorrect simulation runs has dropped considerably. The customer has also found that the simulation runs can now be prepared by relatively untrained personnel, since they do not have to format the file and supply various command characters. The net result was improved reliability and efficiency.

Another example of the use of front and back end code is Tessella's work on enhancing both the interface and data visualization components of a pipeline simulation program. This was a very complex simulation code that took a large amount of input in the form of complex text files and generated a vast amount of data that had to pass through a post processor before any results could be displayed in a meaningful manner.

The customer had invested a lot of effort, over many years, on the FORTRAN simulation code. Their aim was to provide the system with a far simpler means of supplying the input data and an easy to use visualization tool. Tessella chose to sandwich the FORTRAN code in a Microsoft Visual Basic front and back end. The front end component gave the users a typical Windows style interface through which they could supply the relevant variables and constants for the simulation. The back end provided a number of ways to display the results of the simulation, with a graphical representation of the pipe and significant data, being the most important. A typical graphical representation of the pipe is shown in figure 5.

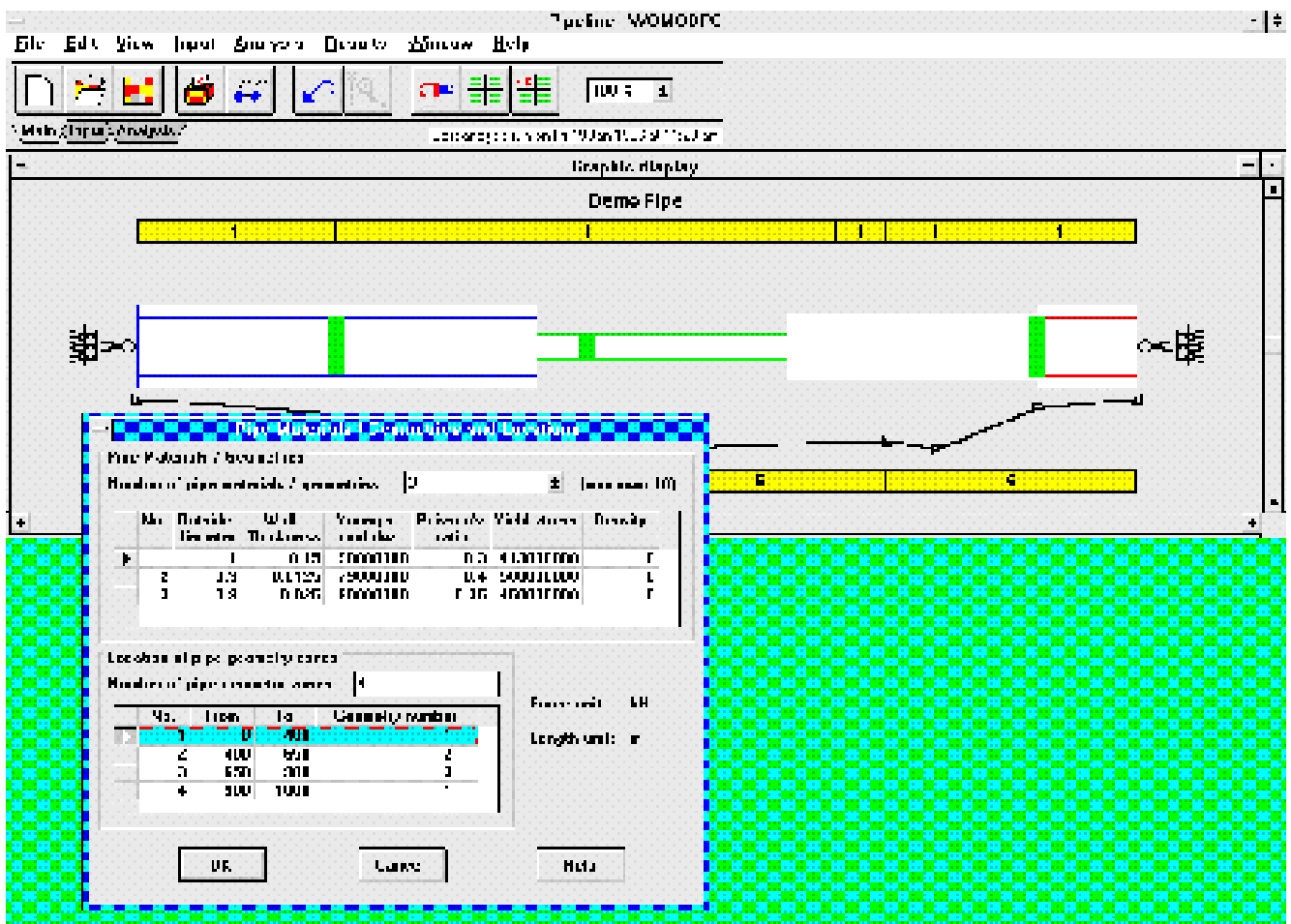


Figure 5: A typical screenshot of the pipeline simulation output.

The customer has found that the simulation is now far easier to use and that less experienced users can generate reliable results consistently. They have found that the instances of data errors have dropped considerably and the time taken to produce results has been reduced by nearly 50% through the use of the data visualization tools.

Develop Interface Clients

This approach tends to be suitable on those sites that make use of mainframe platforms to run their legacy systems. The client-server approach is used because it is often easier and more cost effective to create sophisticated visualization applications on attached PCs or workstations, rather than try to develop these applications on the mainframe itself.

The mainframe operating systems have substantial strengths in security and the ability to manage vast amounts of data using sophisticated hierarchical storage management software or database systems. The client PC and workstation are well suited to performing data visualization and analysis via cost effective graphics packages and graphics accelerators. This approach allows the user to take advantage of the strengths both platforms have to offer.

Tessella has implemented such a solution for a large research site. They have a substantial archive of experimental results held on an IBM MVS mainframe, with users accessing the data via IBM graphics terminals. The aim was to allow PC or workstation users to gain direct read access to the data so that they could process it using tools such as MathCAD, NAG, PV Wave or SAS. This would allow them to use the full graphics potential of the client machine to gain the most from the experimental data.

This solution required the use of TCP/IP on the mainframe and the clients, with the clients running Microsoft Windows 9x/NT. Tessella developed both the server and client software providing the clients with access to about half a Tbyte of experimental data. For this application Tessella provided the users with the pre-processed data and the users then developed their own data analysis applications using pre-defined interfaces to the client software.

The Complete Rewrite

At first glance this may appear to be the least likely approach for all but the most modest application, since it appears to involve a considerable amount of effort. Similarly it appears unsuited to applications that make use of novel algorithms or complex simulations, since these would need to be recoded. There are, however, situations where this would be a very practical and cost effective solution.

A complete rewrite may be the best solution where the site wishes to upgrade the underlying application, e.g. to bring the application in line with changes in

business practices, as well as improving the user interface. In this situation you effectively tackle both tasks at the same time, i.e. you replace the underlying system and enhance the user interface as part of the same development programme.

There are a number of advantages to approaching the problem in this way. Firstly, by developing the new user interface at the same time as the underlying code you will be able to set up a robust and future proof interface between the two components. In this way the user interface can be enhanced or changed at any point in the future with the minimum of effort. Additionally, you will be able to tune the system to the user needs, e.g. obtain the data in an order more suited to the users than the application. Finally, the rewrite will also allow you to review all aspects of the application and to enhance its performance and functionality, giving you the opportunity to produce more detailed or sophisticated analysis of the data which in turn can be inspected via one of the many data visualization tools now available.

As an example, consider the following rewrite of a consultants register performed by Tessella. The original application was written a number of years ago in Computer Associates Clipper to run under DOS. It was recently decided to bring the interface to this application up to date and give it a standard Microsoft Windows look and feel by using the facilities of Microsoft Visual Basic. In addition to the interface upgrade it was decided to improve the performance of the underlying database, and as a result, we dramatically reduced the number of database tables from 14 to a more manageable 3, and also converted from Computer Associates Clipper to Microsoft FoxPro. The net result of these changes was a complete rewrite of the application.

Figure 6 shows a typical user interaction with the database. This was formerly achieved by the user typing the relevant details in a line mode table.

Conclusions

Many of our customers have found quantifiable business benefits by upgrading the interfaces to their legacy systems. Improving front ends, and adopting a standard user interface for all software not only results in an increase in the potential number of users able to access each system, but also reduces the time spent on user training and minimizes their subsequent supervision by so called *application experts*.

Tessella Support Services plc
Creating Software for Science and Engineering

Tessella's services range from feasibility studies, through system design, development, implementation and ongoing support. Our expertise includes:

Data Analysis Software
Data Capture
Simulation Software
Advanced Graphics
Systems Support
Database Applications

Other Technical Supplements available include:

- | | |
|---------------------------------------------------------|--------------------------------------------------------|
| <input type="checkbox"/> Archiving of Electronic Info | <input type="checkbox"/> Object Oriented Programming |
| <input type="checkbox"/> Active Server Pages | <input type="checkbox"/> Pocket PC |
| <input type="checkbox"/> Automated GUI Testing | <input type="checkbox"/> Portable GUI Development |
| <input type="checkbox"/> Bayesian Statistics | <input type="checkbox"/> Printer Technology Guide |
| <input type="checkbox"/> Beowulf Clusters | <input type="checkbox"/> Real Time Systems |
| <input type="checkbox"/> C++ | <input type="checkbox"/> Regression Testing |
| <input type="checkbox"/> Client-Server Technology | <input type="checkbox"/> Security and the Internet |
| <input type="checkbox"/> COM | <input type="checkbox"/> Simulation |
| <input type="checkbox"/> Computational Fluid Dynamics | <input type="checkbox"/> Soft Computing |
| <input type="checkbox"/> Computer Image Processing | <input type="checkbox"/> Software Design Methodologies |
| <input type="checkbox"/> Decision Support Systems | <input type="checkbox"/> Software Development Cycle |
| <input type="checkbox"/> Electronic Data Capture | <input type="checkbox"/> Software Documentation |
| <input type="checkbox"/> Electronic Lab Notebooks | <input type="checkbox"/> Software Portability |
| <input type="checkbox"/> Excel | <input type="checkbox"/> Software Re-engineering |
| <input type="checkbox"/> Extending the Life of Software | <input type="checkbox"/> Software Specification |
| <input type="checkbox"/> Federal Drug Administration | <input type="checkbox"/> SQL |
| <input type="checkbox"/> FORTRAN 90 | <input type="checkbox"/> UNIX Inter-Process Comms |
| <input type="checkbox"/> Grid Computing | <input type="checkbox"/> UNIX Systems Performance |
| <input type="checkbox"/> High Throughput Screening | <input type="checkbox"/> UNIX Workstations |
| <input type="checkbox"/> Instrumentation | <input type="checkbox"/> Visual Basic 6 |
| <input type="checkbox"/> Integrated Lab Systems | <input type="checkbox"/> WAP |
| <input type="checkbox"/> J2EE | <input type="checkbox"/> Web Services |
| <input type="checkbox"/> Java | <input type="checkbox"/> Windows 2000 Services |
| <input type="checkbox"/> Lims | <input type="checkbox"/> XML |
| <input type="checkbox"/> Linux | |
| <input type="checkbox"/> Microsoft Net | |



INVESTOR IN PEOPLE

Tessella Support Services plc

3 Vineyard Chambers, Abingdon, Oxon, OX14 3PX, England

Tel: (+44) (0) 1235 555511 Fax: (+44) (0) 1235 553301

E-mail: info@tessella.com Web Address: <http://www.tessella.com>