METHODS & TOOLS

Global knowledge source for software development professionals **ISSN 1023-4918**

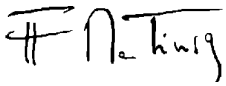
Fall 2000 (Volume 8 - number 3)

Can I Get Some Training?

Training courses come in many forms. You got the one week components training with its bonus trip to the Côte d'Azur or to Florida, the internal three days course on SQL standards, the half-day seminar to understand everything about UML or the two-hour video on Java or e-commerce. Even if every brain exercise is a good thing, how many of these training courses are really useful for your day to day activities as a software developer or a project manager? How many good training will you find on configuration management, designing application architecture, software testing or negotiating with the end user?

And how many organisations are just giving their developers the training they need? According to our evaluations of software development units, one organisation out of three gives its software developers less than one week of training per year. One organisation out of two gives its software project managers less than one week of training per year. Could this be an explanation for the poor rate of success? Excuses for not having training are always found... more often that the budget for getting it. There is always something more important to do or your manager will find that the course is not related to your current work. How can you progress this way? Developers and project managers want to learn more. They are even ready to go through editorial like this one to reach some practical knowledge!

I would like to contribute to the debate with some basic data. Companies that give more training to their employees achieve a better process quality. Companies with a higher process quality have more chances to respect the estimated deadlines and budgets of their software development project. So the next time you will have to discuss training budget and availability, and that is often do at the end of the year, bring with you the two graphs included in the piece on training and process quality. It could cost to train, but it also pays!



Inside

Risk-Based E-Business Testing - Part 2 Test Techniques and Tools

Paul Gerrard, paulg@evolutif.co.uk Systeme Evolutif Ltd. www.evolutif.co.uk
9 Cavendish PlaceLondon W1M 9DL, UK
Tel: +44 (0)20 7636 6060, Fax: +44 (0)20 7636 6072

Abstract

This paper describes twenty techniques for testing E-Business applications. These techniques were introduced in the companion paper: "Risk-Based E-Business Testing, Part 1, Risks and Test Strategy". The techniques are described under five categories: Static Testing, Test Browsing, Functional testing, Non-Functional Testing and Large Scale Integration testing. A chapter on Post-Deployment Monitoring is appended. *(Note from the editor: due to size limitation, this part will be split in two. You can read the complete text at: www.evolutif.co.uk/ articles/EBTestingPart2.pdf)*

Static tests aim to identify statically detectable faults in web pages. Tools that can check the syntax of the HTML and other embedded code in web pages are now widely available. Internet browsers and versions implement different subsets and supersets of standard HTML but tools can identify anomalies in the HTML as well as make recommendations for correct usage. Between the leading browsers, there are differences in the appearance of web pages containing identical HTML that has been validated. These differences are often cosmetic, but can affect more than the appearance as objects may be pushed off screen because of different object sizing.

Test browsing aims to ensure that web page navigation and integrated pages operate correctly. Link checking verifies that buttons or text links to other pages work. Object load and timing checks aim to ensure that objects can be loaded and are displayed with an acceptable delay. Transaction verification aims to ensure that server-based components are called correctly, with all parameters passed.

Functional testing covers what might be the 'traditional' test approaches to component, integration and system testing. Browser page testing covers the functionality that executes solely on the browser. CGI component testing covers all server-based components. Transaction testing verifies that the browser/user interface, web server and back end server-based components integrate as a whole. Application system testing covers the functional requirements for the system as a whole. Internationalisation covers the requirements for multilingualism, multicurrency and localisation.

Finally, it is becoming more important that web sites are monitored in production as performance, security and reliability may all falter but users may never report problems. Post-Deployment Monitoring may involve reuse of existing automated tests or one of the increasing number of remote web site testing service providers.

1. Introduction

1.1 Overview

This paper describes twenty techniques for testing E-Business applications. These techniques were introduced in the companion paper: "Risk-Based E-Business Testing, Part 1, Risks and Test Strategy", published in the Summer issue of Methods & Tools. The techniques are described under five categories: Static Testing, Test Browsing, Functional testing, Non-Functional testing and Large Scale Integration.

The Test process Framework Table from part 1 is reproduced overleaf.

	Test Priorities					***Test Types Mapped to Usual Test Stages***				
Test Type	***Smoke***	***Usability***	***Performance***	***Functionality***	***Static/ Dynamic***	***Desktop Development***	***Infrastructure Testing***	***System Testing***	***Integration Testing***	***Post-Deployment Monitoring***
Static Testing										
HTML testing	Y				S	A/M				
Browser syntax compatibility	Y				S	A				
Visual browser validation		Y			D	M		M		M
Test Browsing										
Link checking	Y				D			A		A
Object load and timing		Y	Y		D			A		A
Transaction verification	Y				S	A/M		A/M		
Functional Testing										
Browser page testing	Y				D	A/M				
CGI component testing	Y				D		A/M			
Transaction Testing				Y	D			A/M		
Application testing				Y	D			A/M		
Internationalisation		Y			D	A/M		A/M		
Non-Functional Testing										
Configuration testing	Y				D	M		A/M	M	
Performance			Y		D		A	A		A
Soak Testing/reliability	Y				D	A	A	A	A	
Availability					D					A
Usability		Y			S/D			M		
Security				Y	D		A/M	A/M	A/M	A
Large Scale Integration										
External links/legacy system integration				Y	D		A/M		A/M	
End to end functionality	Y				D				A/M	A

Table 1 – E-Business Test Process Framework

1.2 Scope

The paper does not dwell on the types of testing that have direct counterparts in client/server or host-based systems.

This paper only covers systems that have been implemented in the most common technologies, namely:

- HTML static pages
- Embedded VBScript and JavaScript
- Server-based CGI programs, scripts or Active Server Pages.

It does not cover the testing of the following specific technologies:

- Java Applets.
- Java Servlets.
- ActiveX components.
- COM/DCOM server-based objects.

The paper does not cover the testing of large applications being implemented in, say, Java applets. In these cases, the applets themselves may be large applications. These should be component tested in the same way that non web GUI applications are. We suggest that Transaction testing and Application System Testing are as appropriate for Java applets as any other application implemented using Web technology.

Many of the test types described in the following section justify a paper of their own. Performance, security and usability testing are each significant topics requiring through treatment.

Most sections provide references for further reading.

One of the challenges of E-business testing is: where does developer testing stop and higher level testing begin? It is probably appropriate that the test techniques described in the early sections are done by developers rather than system or acceptance testers. Allocation of responsibility for more and less technical techniques is a major consideration for your test strategy.

2. Static Testing

Static tests are those that do not involve executing software under test. Usually, static tests involve inspections and reviews of documentation, but can also cover static analysis of code using automated tools. In the case of web sites, we are most interested in automated methods for inspecting and validating HTML for syntax faults but also for compatibility with the various browsers. Although tools can validate the underlying HTML, we recommend that you also perform a visual check to ensure (mainly cosmetic) differences between browsers do not cause problems for users.

Tools that perform automated HTML validation and compatibility often perform other functions such as link, spelling and accessibility checks. These tools are now widely available, some are free, others can be obtained at a small cost. Some tool vendors bundle this functionality with their proprietary test tool suites.

There are four main areas of concern, that static testing aims to address.

Content checking

The content of Web pages, the accuracy, completeness, consistency, spelling as well as accessibility are no longer ‘cosmetic’. It could be said that cosmetic does not equate to unimportant, as perhaps it might for internal IT systems. For many E-Business sites, content might be their ‘raison d'être’. For example, if a Testing Services company spelt "kwality" wrong on their home page, this might not reflect well on the organisation. If content itself is the product for sale or is intended to be reused by customers, content faults might be deemed serious threats to success of the system and should be checked.

Figure 1. - Typical Web page validation tool

Most commercial web page development tools include spell checkers. Just like any documentation, however, pages should be independently reviewed to ensure they are consistent, grammatically correct and usable. This task might be included in a usability test.

HTML testing

Browsers download HTML pages from Web servers, interpret the HTML text and render the page within the browser window. In most ways, this is a similar process to software interpretation. However, unlike software interpreters, browsers are very forgiving of poor quality code. Invalid HTML syntax is usually ignored by the browser. In some ways, this is good. There are no unsightly and confusing syntax error messages displayed, ever. But if the programmer has got the HTML wrong, they may never achieve the layout effects or functionality they require. Web page developers and testers may not see problems, but users may experience obscure difficulties later.

Automated tools can validate HTML (or XML or WML) against rules laid down by established and emerging standards.

Tag	Attribute	Lines	Visitors Affected	Microsoft			Netscape		
				3	4	5	2	3	4
A	ALT	139	99.00%	N	N	N	N	N	N
BODY	LEFTMARGIN	10	27.00%	Y	Y	Y	N	N	N
BODY	TOPMARGIN	10	27.00%	Y	Y	Y	N	N	N
FONT	FACE	14,23,25,29, 33,35,37,38, 39,45,46,48, 51,55,56,58, 63,64,66,69	1.00%	Y	Y	Y	N	Y	Y
IMG	</TD	149	99.0%	N	N	N	N	N	N
TD	BGCOLOR	21,24,63,10 1,110,134		Y	Y	Y	Y	Y	Y

Figure 2. - Browser compatibility report

Figure 1 is the HTML Toolbox page of the NetMechanic web page checker (reference 4). This is typical of the web page test portals now available. As you can see, it can check the following:

- Links
- HTML syntax check (NetMechanic can also generate corrected HTML for you)
- HTML Browser compatibility
- HTML load times
- Spell check. (You can supply your own dictionary, rather than use the one supplied).

Browser syntax compatibility

Although HTML is often promoted as a universal standard (Reference 26), browsers actually implement HTML differently. Some elements of HTML from previous standard are deprecated in HTML 4, but are still supported by these browsers as they are in common use. The event model (that defines behaviour for mouse clicks, button presses etc.) is far from complete in the current Standard, compared to what Visual Basic, for example, supports, so Microsoft, in particular, have implemented an enhanced event model for Internet Explorer.

Microsoft Internet Explorer (IE) and Netscape Navigator (Netscape) are clearly adopting different strategies:

- The IE strategy is to broadly support the HTML 4 standard, but Microsoft have introduce many additional elements that are supported by their browser and generated by their popular web page development tools (FrontPage and Visual Interdev).
- The Netscape strategy is to broadly support the HTML 4 standard, but Netscape have introduced a much smaller number of non-standard elements. It appears likely that Netscape will drop the deprecated features they have introduced.

A typical report on web page HTML compatibility appears in the figure 2. You can see that the ALT attribute on an anchor tag (<A>) on line 139 of the web page is incorrect for both Explorer and Netscape. The LEFTMARGIN attribute of the BODY tag on line 10 is valid for all versions of Explorer and invalid for all versions of Netscape.

Setting aside the fact that the Microsoft strategy for browser domination appears to be working, in the interim, the existence of these two strategies causes difficulties.

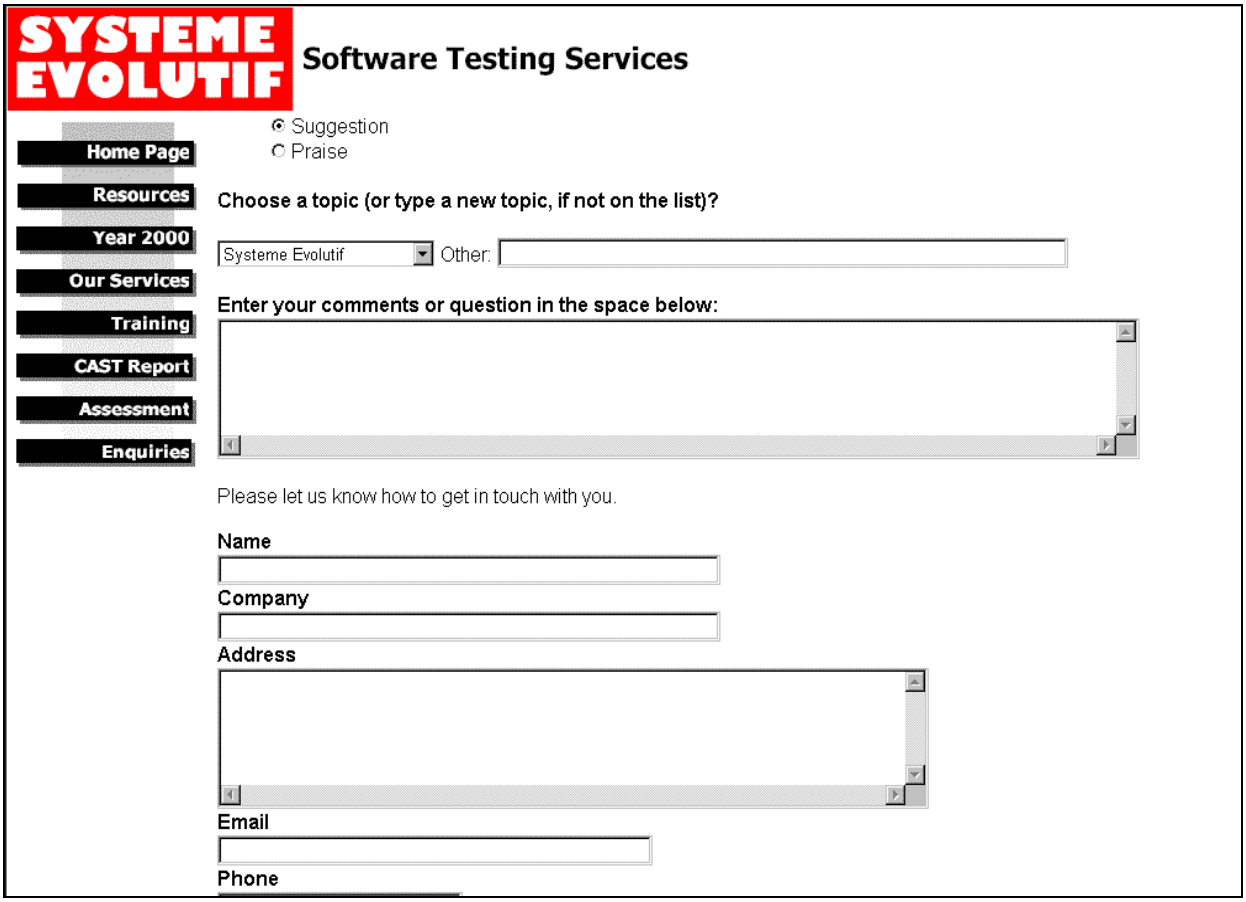


Figure 3. - Web page display, first example

To support both browsers simultaneously, there are three approaches:

1. Adopt a reduced set of HTML elements within the HTML standard, that are supported by both browsers and restrict yourself to those.
2. Build parallel HTML implementations for each browser and determine which should be used at runtime. (Potentially complicated and more costly)
3. Adopt one browser as standard and ignore all others. (Only viable if you are building an Intranet and have control over your users' technology platforms).

Page authoring products usually have configuration options to allow HTML that meets one of these following compatibility requirements:

- IE only
- Netscape only
- Both IE and Netscape
- Web TV
- An so on…

Once a strategy has been selected, you must agree with the developers whether (and how) they are adopting this strategy. Then you must use the available tools in a consistent way to report faults.

Visual browser validation

It should be obvious from the previous discussion that there will be significant differences in the appearance of web pages displayed by the different browsers, but also in their behaviour. Take for example the two identically sized screen dumps of a previous version of the Evolutif Web site. One is from IE4 and the other Netscape 4. (I must admit, I cannot recall which is which!).

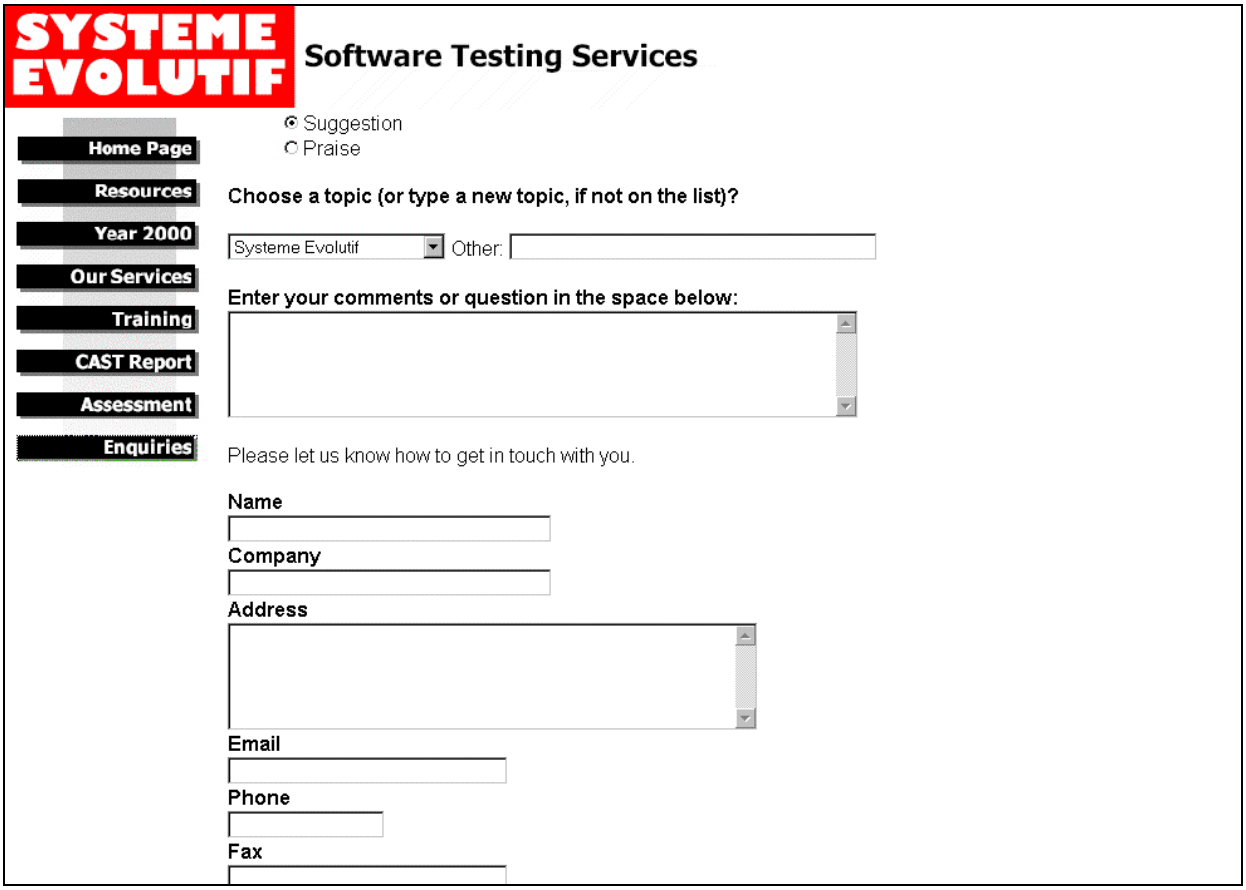


Figure 4. - Web page display, second example

Note that the last field displayed on the first page, displayed in figure 3, is 'Phone' – and the field itself is almost completely cut off by the border of the page. In the second example, displayed in figure 4, the last field is 'Fax'. In this case, an extra field is displayed by the browser.

Clearly, the differences here are cosmetic, but if your intention is to avoid scrolling of pages in a set resolution setting, then this problem might force a clickable button off the page and be a less usable. Other, more serious problems have been reported, but are most commonly associated with differences in:

- The colours of standard objects e.g. horizontal lines, borders etc.
- The centering and scaling of objects in windows
- Display of scrolling marquees
- Table layouts - automatic sizing gives different results
- Table background and border colours differ

It would be possible for a cross-browser automated tool to be used to scan pages for examples of page objects being displayed differently, but we strongly recommend that you adopt a more pragmatic approach. Consider testing under one platform and verifying visually, that the appearance of pages on the other browser are acceptable. We recommend that your developers (and testers) do not all use the same browser and version. Let them use their favourite browser or ask them to use a selection of browsers and versions, to avoid compatibility testing.

3. Test Browsing

Test browsing aims to address those faults that relate to the navigation through web pages, the availability of linked objects and

the speed of download. It also covers the integration of web pages to server-based components, to ensure that the right component is called with the correct parameters.

3.1 Test browsing

If we were to focus on traversing links and opening pages, when we open a new page:

- Can the page be downloaded and displayed?
- Do all the objects on a page load?
- Do all the objects on a page load in an acceptable time?
- If the user turns images off, uses a non-graphical or no-frames browser, does it still work?
- Do all the text and graphical links work?

One of the most common problems with web sites relates to validity of on-site of off-site links. On-site links should load pages or objects from the same site of course, but many sites, especially search engines or directories have many, many links to off-site pages, normally other sites.

3.2 Automated browsing

There are many tools that can perform automated browsing tasks. Automated browsing covers the first three tasks, link checking, object loading and timing. Checking that pages work when users turn graphics off or use of no-frames browsers must be done manually.

Link checkers read a starting page and identify all the linked objects on that page. These can be any of the following:

- Linked pages (other pages to be navigated to by clicking on hyperlinks).
- Frame pages (where a page is partitioned into frames and each frame has its own HTML page to create the image displayed in the browser window.
- Images used for graphical appearance or as buttons to navigate (e.g. GIFs and JPEGs)
- Form handlers, where these are CGI scripts, Active Server Pages etc.
- ActiveX, Java applets and other objects that are downloaded and executed within the Browser.
- Other content files, such as video (AVI, MPEG), and audio (WAV, AU, MIDI, MPEG) files.
- Other Internet protocols such as email links, FTP, Newsgroups links.

Typically, the tools that do link checking perform two functions. Firstly they identify all of the linked objects on a page and then attempt to download them. In doing so, the robots determine whether the object exists and can be loaded, and the size and download time for the object. Typical reports list objects linked, validity and size. Some tools provide download time for each object-based on a selected network connection speeds. Nowadays, a typical modem connects at 33kbps. This is a reasonable assumption for most business-consumer applications. For business-business applications, end users might be connected at much higher speeds using their corporate LAN.

These tools work either as a portal-based services, where you request a Web robot or agent to visit your site on the web and traverse all of the links and objects and then produce a report.

Some tools provide graphical output of the links for a site with the home page at the centre of such graphics. These diagrams normally resemble a collection of connected spidery shapes. This is less useful for large sites (although the diagrams can be pretty!)

The majority of Web page development tools, e.g. Microsoft FrontPage, offer link checking facilities. In effect it is simple the object file size and connection speed that determine the download speed so these tools

can usually report broken links and slow download speeds.

3.3 What is an acceptable download speed?

Firstly, you need to consider who your users are. In general, object sizes are not critical for users who connect using a 2 Mbps or T1 line (1.544Mbps) direct to the net or to our local Intranet. You must bear in mind that home-based and small office-based users may be accessing the Web through 28k or 56k modems. You remember accessing the web using your 9600bps modem, don't you? In those days, you probably turned graphics off entirely. Nowadays, the standard modem speed is around 33k. This is what you should use for assessing the speed of downloads. Until new technologies like ADSL are established, the size of pages will remain a big issue for consumer users.

The main concern with speed of page loading is page design, not the speed of our infrastructure and the net itself. Normally, if page graphics are not the main content we are offering on a web site, the images used to illustrate our site and provide graphical buttons for navigation and transactions are our main concern.

Normal recommendations for good (and fast) page design are:

- HTML IMG tags have WIDTH and HEIGHT attributes to define image size explicitly.
- The home page should be less than 30k in size total.
- All other pages less than 45k.
- Background and button/icon pictures should be less than 5k.
- If you must use an image for the page background, use the same image for all pages.
- Photos are JPEG format, computer generated images/schematics should be GIFs.
- All GIFs should be interlaced.
- Images should not be duplicated – use the same images repeatedly to avoid reloads.
- Limit table text size to 2k.
- Size tables explicitly rather than use browser autoformat.

3.4 Transaction verification

The most common mechanism used to implement functionality on web sites is the Common Gateway interface or CGI. When an HTML form is filled in and the form submitted using one of the form buttons, the form handler referenced by the form definition is invoked on the server. The form itself sends the data captured on visible fields and the preset information defined on invisible fields as parameters to the server-based form handler.

Server-based forms handlers are CGI programs. On Unix systems, these are Perl scripts or executable programs written in C++ perhaps. In the Windows environment, these programs can be C++ programs, DLLs or Active Server Pages.

What happens is that the forms handler extracts the parameters supplied in the message sent by the browser and uses this data to perform the server-based transaction. Typically, forms handlers execute query or update transactions on a database and then generate a new page to be posted back to the client browser.

Transaction verification aims at ensuring that firstly, the correct forms handler is invoked and that the parameters passed to the forms handler are correct.

Part of the forms definition is the method by which the HTTP message is sent to the server. There are two methods used: GET and POST.

HTTP GET

When a CGI program is invoked by an HTTP GET message, the parameters are appended to the URL of the forms handler. For example, a form having two fields, reptype and sortorder, that generates a report would generate a URL such as:

```
http://www.mysite.com/cgi-bin/
doreport.pl?reptype=ProdList&
sortorder=ByPrice
```

In this case, the program doreport.pl, in directory cgi-bin on the web server would run with two parameters, reptype is "ProdList" and the sortorder is "ByPrice". The full URL above, including the parameter values is displayed in the URL window of the browser.

HTTP POST

When a CGI program is invoked by an HTTP POST message, the parameters are incorporated into the body of the HTTP message (not in the URL, like GETs). The browser does not display the form parameters appended to the URL in the URL window. The forms handler obtains these values from either environment variables or the standard input to the program. In this case, in order to verify that the parameters are passed correctly, the programmer would need to incorporate a small amount of code to write out the parameters passed to the forms handler program. This code would be commented out when any anomalies are fixed and the code is working correctly.

4. Functional Testing

Functional testing has been treated as a staged, bottom-up approach. Components that execute entirely within the browser are tested separately from the server-based components. Transaction testing addresses the integration of the browser pages, web server and other server-based components. Application System Testing addresses the traditional requirements for a complete functional system test. Internationalisation checks cover the particular requirements relating to address formats, multi-currency and tax arrangements.

4.1 Browser page testing

Browser page tests cover the objects that execute within the browser, but do not exercise the server-based components. These are typically:

- JavaScript/VBScript code within HTML pages that:

  Implement special effects, such as changes to the appearance of buttons as the mouse pointer rolls over the button.

  Implement field validation.

  Open new windows, control the sizing and positioning of windows etc.

- (Usually Java) applets that implement screen functionality or graphical output.

These features of the web pages should be tested in-situ using the browser. These features are most easily tested against checklists of 'conventional' requirements or the specific requirements for particular pages.

NB: Some systems implement a large amount of functionality in Java applets; some entire applications are implemented in applets. In these cases, test them like any other GUI application. For advice on testing GUI applications, see reference 8.

4.2 CGI component testing

CGI component testing covers the objects that execute on the server, but are initiated by forms-based user interactions on the browser. These are typically:

- Business logic to do database enquiry or updates.
- Product catalogue searches.
- Order processing.
- Credit checking and payment processing.
- Security checking and validation.

```
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=BookList"
target="_test">Book Listings</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=Event"
target="_test">Events, Conferences Etc.</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistbooks.asp?resType=BookStore"
target="_test">Evolutif Tester Book Store</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=Forum"
target="_test">Forum/Newsletters</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=FAQ"
target="_test">Frequently Asked Questions</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=Guideline"
target="_test">Guidelines</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=OnLineTraining"
target="_test">On-Line-Training</a><p>
<a href="http://ntofclose/Evolutif/cgi-bin/tlistresources.asp?resType=Organisation"
target="_test">Organisations</a>
```

Figure 5. - HTML source code in the test driver

Usually, the server-based forms handlers are written at the same time as the forms that invoke them. However, we have found two problems with testing server-based code via the user interface:

- Firstly, in some projects, the user interface is actually the last thing to be written so server-based components cannot be tested completely until very late in the project.
- Secondly, it is common for server-based components to be programmed to deal with a very broad range of input parameters, invoking complex transactions that interface with legacy systems. Large numbers of tests may be difficult to implement manually so an automated alternative can be more effective and economic.

We advocate the use of dummy web pages or test drivers to test server-based components. In this way, the manual effort can be leveraged dramatically. Clicking on one of the links of the dummy page window sends the HTTP message to the web server and displays the results in another window. When a new link is clicked, the results' window displays the results of the transaction initiated.

The HTML source code for the dummy page is presented in figure 5. The anchor tags (the <A> </A> pairs) implement HTTP GETs of the URL identified in the HREF attribute. Note that the display of the full URL in the URL field of the browser window on the right.

If you look closely at the first line of HTML in figure 5, this generates the underlined link labelled “Book Listings” in dummy window. When you click on this link, the browser sends an HTTP GET message, referencing the following URL:

```
http://ntofclose/Evolutif/cgi-bin/
tlistresources.asp?resType=BookList
```

The resulting page generated by the ASP file is displayed in the window on the right. The internal name for this window is “_test” (see the “target=” HTML).

It is easy to click on each link in turn and verify that the correct page listing appears in the other window. Consider how you might generate this HTML from a simple database containing just two columns: URL and parameter resType. We have used various techniques including mail merge and writing Visual Basic routines within an Access database to generate simple test drivers using this approach.

Alternative techniques involve generation of dummy forms – many, many forms can be displayed serially in the same web page. Each form contains the fields that would have been supplied using the real web page in the application, but is generated by a

database. With forms you can implement both HTTP GETs and POSTS, and in this way, simulate both types of transactions in a realistic, but simple way. There are some issues relating to the set up of cookies and potentially application and Session variables on the web server, but your developers should be able to create simple server-based scripts to ensure these are correctly set up. This is a more advanced topic, outside the scope of this paper.

Clearly, to implement such facilities you need some detailed knowledge of HTML and the way forms, HTTP GETs and POSTs work. However, if you can get the help of your developers or bring a developer into the test team, it should be a simple task to build some very effective test drivers for your server-based code.

In our own projects, Evolutif have built simple automated test drivers using Visual Basic and Access databases. In these cases, we used the Visual Basic INET object to simulate HTTP requests and retrieve web pages and execute server-based code. In this way, we have run functional tests that execute 40,000 transactions in a single test run.

Creating and running a manual test of 40,000 transactions is impractical. Creating and running such large tests using proprietary test execution tools that drive the browsers would also pose a significant challenge. There is a clear choice in this area:

Proprietary tools

Proprietary tools driving the user interface are expensive, easy to use, generate complex GUI scripts and force the tester to navigate slow, complex application scenarios.

Home grown tools/test drivers

Home-grown tools driving server-based code via the API can be cheap, easy to use, generate simple HTTP oriented scripts and eliminate slow navigation through complex application scenarios. They are less good at verifying actual results against expected results.

We strongly suggest that you investigate the use of test drivers for at least the higher volume test transactions as it may save you a lot of time. Further, by involving the developers, you may find that the burden of functionally testing server-based components may shift towards the developers (a good thing!). System testers can perhaps re-use the developers' test drivers to explore more complex and higher volume scenarios.

4.3 Transaction Testing

Transaction testing aims to address the problem of integration of the complete end-to-end functionality to ensure that the entire transaction is processed correctly, from the user action on the browser interface through to back-end systems. Transaction testing is not the same as Application System Testing. Rather it focuses very much on selected test cases that exercise particular interfaces between components in the technical architecture and scenarios (in volumes) that System tests might not address.

Typical tests here would be selected transactions that invoke several components and interfaces that must collaborate to deliver a piece of functionality. Transaction Testing maps to traditional integration testing in the small, where a member of the programming team conducts tests to ensure the products of more than one programmer integrate, before submitting integrated components for system test.

A transaction that processes the payment for a book, bought online might involve the following:

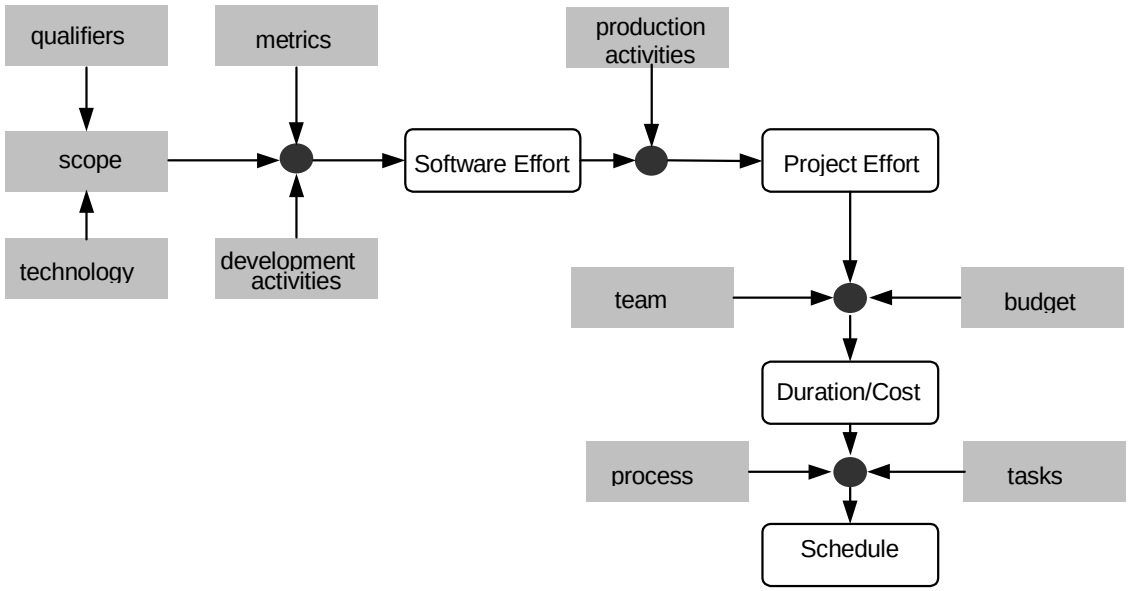
- A button click on the user interface prompting validation of the fields on the screen.
- Dispatch of an HTTP message to the web server, with visible and hidden fields on the user screen being transmitted.

software development is your business

- how long will your development project take?
- how much it will cost?
- **how do you know?**

software development is our business

- how many people have you got to do the job?
- what do you need to build?
- **Optimize can tell you**



successful projects

depend on balancing
- **budget**
- **resources**
- **deadlines**

need to accurately estimate
- **cost**
- **effort**
- **duration**

optimize

the new standard for planning, estimating and controlling software development projects

brings **you** success

- track and control your project by defining phases, iterations and tasks
- enter project scope in the UML manually or automatically import from the world's best CASE tools
- account for non-software engineering activities including multimedia elements, help systems, user manuals
- allocate individual resources to specific tasks - enter individuals' costs, availability and skill levels
- instantly assess the effect of change in scope, resources or budget

download your FREE evaluation software and technical papers

www.**theobjectfactory**.com

- Invocation of a CGI program on the web server that sends requests to various objects such as:
- Credit checking object.
- Order posting object.
- Database update object.
- Special offers and order confirmation email generator object.
- Thank-you web page generator object.
- The response of the CGI program would be a 'confirmation' page with options to place more orders or log off.

The objects called by the CGI program may each have been tested in isolation. Often, they have been tested as they were coded with stubbed out interfaces and simple driver programs or dummy web pages to exercise them in artificial circumstances.

Transaction tests are designed to force the software to invoke the various components as a complete set and to investigate whether the direct and indirect interfaces work correctly. These test cases cover the following main concerns:

- Transfer of control between components.
- Transfer of data between components (in both directions)
- Consistency of use of data across components.

The objects called by the CGI program may each have been tested in isolation. Often, they have been tested as they were coded with stubbed out interfaces and simple driver programs or dummy web pages to exercise them in artificial circumstances.

Transaction tests are designed to force the software to invoke the various components as a complete set and to investigate whether the direct and indirect interfaces work correctly. These test cases cover the following main concerns:

- Transfer of control between components.
- Transfer of data between components (in both directions)
- Consistency of use of data across components.

Typical test cases involve:

- Single shot transactions that pass zero, 1, many or a very large number of records.
- Reconciliation of pre-test database content, data entered through screens, system outputs and post-test database content.

Overall, Transaction tests aim to cover the complete end-to-end functionality within a system including the browser web pages, web server-based objects, other server-based objects, back-end databases and legacy systems.

Often, the interactions between components are complex or there is a significant amount of custom-built code in the server-based objects or there is custom-built middleware involved. In these circumstances, it is prudent to automate some of these transactions to ensure that these interfaces and custom code can support repeated use for an extended period. Repeated transactions driven by a test execution tool (or possible re-use of drivers mentioned above) can expose obscure memory leaks or synchronisation problems.

4.4 Application System Testing

Application System Testing aims to cover complete testing of the application with business oriented scenarios to ensure all features of the system meet their requirements. In this regard, AST is simply System Testing for E-Business systems.

Each feature of the system is identified in turn and test conditions selected to exercise the key functionality. Test scripts might be oriented towards running all of the tests for each feature repeatedly and working one's way through all features, but some percentage (the higher the better) of tests

should follow business scenarios. There is very little more to say about this that is unique to E-Business systems. However, there are two particular points that are worth emphasising.

- Applications can be navigated using the browser using the navigation buttons (these are not under the control of the application).
- The Web is 'stateless'. There is no persistence between transactions unless developers implement certain mechanisms.

Browsers allow the use of back, forward and refresh buttons. These buttons may cause server-based transactions to be repeated, perhaps many times over. How do these affect the use of your application? Is it sensible to rerun the same query twice? Possibly. Is it sensible to execute the same payment transaction twice? Certainly not! How will the application deal with these situations if a user, unwisely, chooses to do so?

The Web is 'stateless'. That is, each HTTP message received by a Web server is unique and unless certain mechanisms are adopted, applications cannot maintain the context of business transactions. What this means is that a web server cannot link a transaction to add products to a shopping cart to a later transaction that pays for the goods unless one of several mechanisms are used. How are the developers designing their applications to maintain the context of business transactions?

Cookies

Because there is no persistence between CGI transactions, the CGI mechanism allows the use of cookies. Cookies are small amounts of data that may be stored on the users hard drive by the browser at the request of the Web site being accessed. When the user clicks a button and sends a new HTTP request to the web server to process a transaction, the cookie data is embedded in the HTTP message and the server can extract this cookie data for its own purposes.

Typically, cookies contain personalisation data such as the user's name, the date of their last visit and so on. Often, cookies contain a unique reference to identify a logged-in user session. All transactions posted to the web server during the session are labelled with this session identifier, so the server-based code can 'remember' who they are dealing with. Used sensibly, cookies are harmless enough.

However, the programmer has total control over cookies and could put more sensitive data such as passwords, credit card numbers and more personal information into them. Because of this, browsers offer users the opportunity to be warned of cookies being sent and also to reject them. If a user rejects a cookie, will your application still work? Or will the transactions fall apart because there is no way to string them together?

There also some limitations on the use of cookies and these are prime candidates for test cases. See reference 10 for a thorough description of the CGI and cookie mechanisms.

Hidden fields

The most common alternatives to cookies are hidden fields. When a user completes an HTML form displayed on a web page, the content of visible fields that the user can enter are appended to the HTTP message sent to the web server. However, there is a facility in HTML to include hidden fields within the form that have pre-set values. These values are also sent to the server at the same time, in the same message. Programmers use this facility to include context information, such as a transaction identifier in the forms that are returned to the browser. Because the hidden fields are invisible, the user never notices that data items like these accompany the data that they are entering. It is a simple mechanism, but very effective. There is of course a security

hole here – you can view the HTML source code in the browser, and change it to suit yourself and potentially affect the behaviour of the system under test.

It is essential that you discuss with your developers how they get round the stateless nature of the web and how this might be subverted. This will lead you to a set of tests that could expose faults in the design of the application.

Cookie Testing

Cookies have an expiration date. That is, the programmer sets a date on which the cookie will be automatically deleted from disk by the browser. Ask the developers how they are using cookies and how they set expiration dates. If these dates could be exceeded, consider deleting them in between tests, or setting client system clocks to force them to expire. What happens in the application?

Cookies have maximum size of 4k each, there is a limit of 300 cookies on a client and 20 cookies for a single domain – are your developers exceeding these limits?

Loss of connection

The context of user transactions can be lost through other means. The first, most obvious one is a loss of connection between the client machine and the web. Users with dial-up modems are used to connections timing out or failing. How does you application cope with this? When you lose the connection and/or close your browser completely, cookies having no expiration date set will be lost. Can (should) the user retrace their steps in the application using the history facility in the browser?

Other situations to consider

What happens if a new user of the same client PC uses your application on the same domain? Will they pick up the cookies of the previous user? Will the new user's cookies overwrite the previous user's cookies?

Ask your developers if or how they can accommodate the above scenarios. If they have a means of recovery or avoidance of problems, you should test it. If not, you may have exposed a problem in their design.

4.5 Internationalisation

Is your site intended to be used by, say, English speaking users only? If not, what steps are being taken to make your site multilingual? In principle, localisation testing is all about verifying that all user messages, prompts and output is translated correctly and that the functionality delivered to the end user is identical.

In this case, there may be scope for automating a set of regression tests of the functionality and parameterising the scripts with alternate translations of system input and output. However, if the application has been developed to reuse tables of alternative messages, then it is probably easier to manually inspect the language-specific text tables rather than regression test functionality that does not change across languages.

In our experience, many sites adopt the rather heavy handed technique of hosting multiple copies of translated HTML pages on the same server, and you choose your own language-specific web site. These need more testing and configuration management is probably a bigger headache.

Functional differences between localised web sites should consider at least the following:

- Local variations in tax arrangements, e.g. Value-Added-Tax, purchase tax etc. (in the US, different states have different tax arrangements of course).
- Address formats: in the UK we have postcodes, in France there are departments, in the US there are Zip codes.
- Foreign character sets can be a potential problem. You should recognise that in foreign countries, users may select a

different character set than English (British or US). Russian, Greek, Japanese character sets are dramatically different, aren't they? Will your validation or database settings allow for this?

There are many, many potential pitfalls in international Web sites. You must discuss with your developers and marketers how best to handle these problems.

References

E-Business Testing

1. http://www.evolutif.co.uk/ E-Business Risk-Based Testing, Part 1: E-business Risk and Test Strategy. Paul Gerrard, June 2000.

Tools and tool Listings

2. www.softwareqatest.com/qaweb1.html Comprehensive Web (and other) test tools listing.

3. http://Validator.w3.org Link checker.

4. http://www.netmechanic.com Link checker and other site validation.

5. http://www.htmlvalidator.com/ CSE HTML Validator it user configurable HTML, XHTML, and WML syntax checker available for Windows 95/98/2000/NT. With a built in browser facility (needs IE4 or later), you can browse pages, see the page source code and validate it at the same time.

6. http://www.cast.org/bobby/ Bobby is a free application that will analyse web pages for their accessibility to people with disabilities. It will also find HTML compatibility problems that prevent pages from displaying correctly on different web browsers. Available as a Web-based services and downloadable tool.

Testing Papers

7. http://www.evolutif.co.uk/ Client/Server Performance Testing, Paul Gerrard and Andy O'Brien 1995. This paper describes the end-to-end process of planning. Preparing, executing and analysing performance tests in a client/server environment. Although it doesn't mention the Internet, the principles appear to be unchanged since the paper was written in 1995.

8. http://www.evolutif.co.uk/ Testing GUI Applications, Paul Gerrard, 1997.

Books

9. How the Internet Works, Preston Galla, ISBN 0-7897-2132-5. Best selling introductory text.

10. Webmaster in a nutshell, (five books on CD-Rom), O'Reilly

- HTML: the definitive guide
- JavaScript: the definitive guide
- CGI programming for the world wide web
- Programming Perl
- Webmaster in a nutshell

11. Understanding Electronic Commerce, David Kosiur, Microsoft Press

12. Designing Web Usability, Jacob Nielsen, ISBN 1-56205-810-X. A very readable, well illustrated and up to date book on web usability.

13. "Hacking Exposed: Network Security Secrets and Solutions" - McClure, Scambray and Kurtz ISBN 0 07 212127 0 http://www.hackingexposed.com

Web Resources

14. E-commerce Awareness Forum newsletter, www.mbs-program.com

15. Web Accessibility Initiative: http://www.w3.org/WAI/www.webreference.com/authoring/design/tutorials.html

16. General Web usability. This is Jakob Nielsen's web site focusing entirely on usability issues: http://www.useit.com/

17 Response times: the three important limits: http://www.useit.com/papers/responsetime.html

18. How to conduct a heuristic evaluation: http://www.useit.com/papers/heuristic/heuristic_evaluation.html

19. Software Usability Measurement Inventory [SUMI] & Website Analysis & Measurement Inventory [WAMMI]: www.ucc.ie/hfrg/ & www.acm.org/~perlman

20. Software for Use, Constantine and Lockwood, ISBN 0-201-92478-1. Comprehensive and up to date treatment of usage-centred design with information on all main methods of usability test and evaluation. See accompanying web site: http://www.foruse.com.

Security

21. Internet Security Policy: A Technical Guide, NIST, http://csrc.nist.gov/isptg/html/ISPTG-1.html

22. eCommerce Trust Study, Cheskin research, http://www.studioarchetype.com/cheskin/

23. http://www.cert.org - generic security bug watch run by the Software Engineering Institute – register and get regular advisories on new vulnerabilities in the major OS, Web and browser technologies.

24. http://www.ntbugtraq.com - same as the CERT site above, but for Microsoft NT only.
25. Firewalls Frequently Asked Questions – excellent starting point for this topic, http://www.faqs.org/faqs/firewalls-faq/

Other References

26. http://www.w3.org/TR/html4/ HTML 4.01 Specification, W3C Recommendation 24 December 1999

27. World Wide Web consortium: http://www.w3.org.

Due to size problems, the second part of this article will be published in... two parts :-]
If you can wait until our Winter 2000 issue for the last part of this text, you can have a look at the complete document at:
www.evolutif.co.uk/articles/EBTestingPart2.pdf

How to Sponsor a Successful Project

Barbara Taylor, The Institute for Management Excellence, www.itstime.com
PO Box 631, Silverado, CA 92676

"CIOs still feel besieged," the headline blares. "Top executives are still hounding CIOs about information systems' value for the dollar and IS alignment with business needs," according to a recent *Computerworld* article. CIOs complain that they do not have adequate resources; executives want more value for the dollar. Both sides seem to feel their needs are misunderstood — often blaming their problems on forces outside their control.

Many of the difficulties faced by IS managers and staff result from miscommunication, leading to confusion and unproductive results. High-dollar, mission-critical projects are started with great fanfare, then somehow fail to achieve their objectives. People become disillusioned; fingers are pointed; careers are jeopardised — on the business side and on the IS side.

Business people often wind up with incomplete systems and new programs delivered late, which then are blamed for delays in getting the "truly important" work done. Users do not understand the "black-box" of IS and are often afraid to challenge the wisdom of the technical folks. IS folks tend to want to give the illusions that they are the holy computer gods to whom users should pay homage. Users know they are often dependent on IS to accomplish their job, yet are unable to communicate their

WebBoard 4.0

Quick & Easy Community Building From O'Reilly!

Online forums and discussions are a must in today's Web-based environment. Whether you're building virtual offices of classrooms, corporate intranets, commerce, or online communities, **WebBoard 4.0** can organize, enhance, and improve communication.

- Create a central location for online communications, discussions, and email
- Customize "boards" for your own look and feel
- Run up to 255 boards with unlimited users and 1,000-user simultaneous chat.

It's fast, powerful, and scalable. It's the original WebBoard from O'Reilly. Visit us at www.webboard.com/nl2 and find out why WebBoard is the world's leading discussion and chat software.

http://www.webboard.com/nl2

O'REILLY® WebBoard™
WEB FORUMS & CHAT

needs to the IS folks who systems according to their understanding of users' specifications, which often turn out to be not what the users really wanted. When it comes to new projects, the business folks feel only they have the real handle on corporate priorities and that they should control corporate re-engineering efforts.

IS folks accuse upper management of constantly changing priorities, failing to support the IS folks, seeking outside advice without the participation of IS, not giving the IS manager a seat at the Executive Council, then criticising IS for failing to deliver against corporate goals. IS folks feel unappreciated, that the "plum" jobs go to outside contractors while they are stuck with the unglamorous maintenance chores, and that management takes them for granted. Most IS directors feel that they *do* understand business already, and since they also know about technology they are the best group to lead corporate-wide re-engineering efforts.

Is there a middle ground, or will this constant war of the resources kill any chance for mutual understanding? Is there a way to blend the talents of both the business side of the house and the IS side of the house, so that everyone gains from the experience?

As the move toward more efficient business increases — through the major restructuring of widespread corporate re-engineering efforts and the move to client/server based applications — the need for better working relationships becomes even more important. For the survival of IS and business to utilise the vast benefits possible from new technologies, *it is time to stop the war* and move toward productive teamwork.

One way to bring the two sides together is through the successful management of new projects. All current research shows that the most successful IS projects are those that have good advance planning, good project management and the right people assigned to the project including an *Executive Sponsor*.

From the confusion that still exists between the business side and the IS side of most organisations, there seems to be a viable opportunity for improvement by defining the role of the *Executive Sponsor*. This paper provides some guidelines for the people who may find themselves asked to play the role of *Executive Sponsor*. IS directors can use the list to open a discussion with their business executives — hopefully providing an opportunity for more understanding.

Business executives who read can use the list to open a discussion with their CIO about how the two can better work together.

How to Sponsor a Successful Project

1. Understand what makes projects succeed or fail
2. Know the danger signs of troubled projects
3. Appoint experienced project managers and team members
4. Use outside consultants to supplement, if necessary
5. Provide leadership, guidance and support to the team
6. Enforce the ground rules of good project management.
7. Help the team manage expectations
8. Conduct regular, productive committee meetings
9. Hold people responsible and accountable
10. Don't micro-manage the project
11. Stay out of the way most of the time
12. Know when to step in
13. Know when to ask for help
14. Don't accept guesses — make sure the facts support all project adjustments
15. Identify and solve small problems as they surface

16. Understand the dynamics of your organisation's culture
17. Use your political clout, if necessary
18. Be the project's champion
19. Reward success
20. Set a good example.

Project Success Factors.

There is massive research on what makes projects succeed and fail. Why then do projects continue to fail at alarming rates? Only 16 % of IS projects were completed on time and within budget, according to a 1995 *Computerworld* report. Obviously, there are a few folks who are not reading the research or have missed a few tips along the way.

Some projects start out doing all the right things, then get in trouble. Others simply meander along without any clear direction — "guided" by people hoping that things will somehow fix themselves. Leading projects run like a well-oiled machine — arriving on target, within budget and meeting user expectations.

What makes one project succeed ahead of schedule where another fails miserably?

Known success factors:

1. A project manager with comparable experience in large-scale endeavours
2. An experienced core team supplemented by outside experts
3. A well-defined project plan
4. A rigorous quality assurance process
5. Open, honest communications
6. A process for keeping the team energised and motivated
7. A committed executive sponsor with the time and energy to be the project champion.

Factors that lead to project failure:

1. Incomplete requirements
2. Lack of user involvement
3. Lack of resources
4. Unrealistic expectations
5. Lack of executive support
6. Changing requirements and specifications
7. Lack of planning.

Red Flags.

A key responsibility of the executive sponsor is to provide high-level guidance to the implementation team and to be the project's champion within the organisation. The sponsor must be constantly encouraging the team, yet watching for potential bumps in the road that might de-rail a project.

Indicators of a high-risk project:

1. The project is the largest ever taken by an organisation
2. The project is highly integrated (that is, interlocking systems depend on each other in order to function)
3. The software is new (or relatively new) and has not yet been proven in the market
4. The software needs additional modification to meet your organisation's needs
5. The project uses leading edge technology (affectionately known as *bleeding edge)*
6. The project is out-sourced to consultants who are inexperienced with a project of comparable size and/or unfamiliar with your industry
7. Internal technical staff, key users and/or project manager have not successfully implemented a project of comparable size or complexity.

If *any* of the high-risk factors exist, the project starts with a considerable handicap. Obviously, projects with multiple high-risk factors have a higher handicap.

Attention Software Developers and Process Professionals

A large, undeveloped US and International market need exists in the Process industries for recovery and display of remote information:

- ***Gas production, gathering, processing and transmission***
- ***Chemical plants***
- ***Refineries***
- ***Manufacturing industries***

The ***market opportunity*** lies in management's need to, recover and assimilate information in report and decision making format, the vast amounts of process information which is lying in PLC's, SCADA and various metering, HMI storage devices,

This business ***"NICHE"*** is glaringly evident to the writer, who as a fast growth business developer, is expert in identifying and partnering with technical folks [Specialty software companies, Regularity and Advanced Process Control] to solve process and information control problems.

I seek an International firm that is:

- **Skilled in WEB-based, Real Time, Process Information Technology**
- **Dedicated to US and International growth**

I can develop the market and provide the management assimilation
Know-how at the project level
Please feel free to call me for additional information

John Thackway
***"FIT"* Business Development Processes**
2255 Eldridge Parkway • Suite 1516 • Houston, Texas • 77077
281-647-6120 TEL • 832-483-6995 CEL
E-mail: jthackway@aol.com / jthackway@intechww.com
WWW.FIT-bizdev.com

The executive sponsor may have to supplement the project team (both technical staff and users) with experienced people — whether outside consultants or others in the organisation — who have been through a major project similar in scope to the new project.

A key red flag: individual egos getting in the way of good teamwork. If this happens, the project may have to be stopped until either changes are made or sufficient training is done to insure that people can work together *as a team*. The most elegant hardware and sophisticated system cannot overcome people's resistance to accept the finished product.

Consultants can supplement he project team.

The value of consultants is extensive experience and unique skills gained through many different projects and many different organisations. Often, jealousy arises on the part of internal staff, who feel they have been bypassed by people who are *outsiders*. This may happen in user departments as well as in the IS department. The greatest value of using consultants is when they become *part of the team*, working along side the internal staff and sharing their experience so that when they leave, internal staff has developed new skills.

If consultants do only a specified job (i.e., implement and install a new system) without passing along their knowledge to internal staff, the organisation is weaker after they leave — not stronger. This occurs when the organisation becomes dependent on consultants doing the work *for* them if staff does not take the initiative to improve their own skills as the project progresses. When internal staff does not accept consultants as valued, respected *members of the team*, the entire team suffers. Some consultants actually promote this type of separation by holding themselves aloof and apart from their clients.

This is similar to a baseball team composed of nine players who are more concerned about their individual priorities than the priorities of the entire team. They do not assist each other or provide automatic backup unless forced. A high-energy baseball team functions as a fluid force, where the actions of the individual players flow as a unified movement of the team rather than as individuals who wait for someone to tell them what to do next.

A highly productive project team is possible in the same way. To accomplish this, it takes commitment on the part of internal staff, external consultants, the executive sponsor and the support of the organisational culture. A one-time pep talk at the beginning of the project is not the answer; ongoing leadership example and continuous efforts to encourage and reward team participation are vital to a well-functioning team, just as constant coaching is needed to keep a winning baseball team performing at their peak ability. The executive sponsor should have the human understanding, team-building skills and communications skills as well as the broad organisational vision to be able to prevent such divisions from occurring, or to nip them in the bud should they surface during the project.

Discord in the team continue can destroy the project or lead to crippling cost over-runs due to loss of productivity.

Assign experienced project managers and team members.

The value of having the *right* people on the team cannot be emphasised too much. Successful projects occur because the right people are involved, not because of hardware, software, networks, spiffy screen design or any other technical gadgets. It is always *people* who make the project work. If the people are not committed to the project, to getting the right things done and working together, no amount of fancy technology can make the project succeed.

One of the inherent dangers in technology projects is to become caught up in the technology and ignore the people issues. The executive sponsor must be a people-oriented person first and foremost. If they are able to work with technology issues, so much the better. The sponsor must be an inside coach, constantly promoting the positive aspects of a project to the team as well as helping other executives understand how the new project will benefit them.

Provide leadership.

Leadership is a *fuzzy* quality — people know it when they see it, yet have difficulty describing it exactly. One of the ways IS can help their executive sponsor is to participate actively in regularly scheduled project status meetings with the sponsor. Many organisations have established an Executive Technology Council or Project Steering Committee for this purpose. At these meetings, key members of the team should discuss the progress of the project, identify and resolve outstanding issues, and brain-storm ways to make sure the project stays on target. Each person must be willing to understand the pressures faced by the others, then develop ways to deal whatever challenges arise.

One-on-one meetings between the project manager(s) and the executive sponsor must occur regularly, in addition to the larger group meetings. The one-on-one meetings must be an open forum and a vehicle for developing rapport between the leaders of the project. Trust develops over time. Nothing builds trust between people better than a consistent working relationship devoted to solving mutual problems.

Leadership starts by *setting a good example*. When executives and IS work well together, their staff gets lessons in the right way to do things. For all the good words managers might deliver about motivation, nothing motivates and inspires people better than *seeing a leader in action* and watching how it is done. *Leadership by Example* is the by-word of all great leaders.

Advice to the executive sponsor: If you don't know anything about technology, don't pretend you do — ask IS management for help or ask another executive who is computer literate to help you. If you know a little bit, be willing to learn and try to apply what you learn to your day-to-day responsibilities. Be willing to learn from IS and be willing to share your business knowledge with them. Try to find a middle ground where you can build both your own knowledge and develop a solid working relationship with a technology-savvy person to assist you when you need information.

Advice to IS management and staff: If you think you are the expert on technology and you cannot understand why business people have trouble with technology, you are already in deep trouble with your company management. Be aware than many people are still intimidated by technology; your arrogant attitude will not help you or them; business people constantly complain about IS folks and their inability to speak in anything other than computer-speak. Be willing to explain things *step-by-step* in ways the business people can feel comfortable with you.

For top executives, sending someone to their home or to their office for one-on-one tutoring helps the executive learn by making them feel more comfortable. Find someone who is a good teacher and can earn the respect of executives, then designate them as executive liaison — someone who will not laugh at an executive's simple questions, or think they are stupid for asking. Another approach is to find an executive who is already computer literate to act as a "coach" for those executives who are still technology novices.

Remember, teach executive how to *use* technology in their job, not become techno-wizards. Keep it simple and easy until they ask for more advanced instruction. If at all possible, streamline menus and sign-on procedures to make their technology

experience as painless as possible. They want bottom-line answers, not bytes!

Project management ground rules:

1. Good up-front planning with realistic budgets and schedules;
2. Documentation of needs, goals, deliverables and status;
3. Standard project management methodology;
4. Candid communication at all times
5. Rigorous testing and quality assurance.

Occasionally, the sponsor must provide discipline to the team by setting a high standard and refusing to short-cut the quality controls. Projects often face delays as people under-estimate the work required or find they want to change or add features after they start seeing the prototypes.

Details of a project are often missed in the early stages; people are anxious to get started, rather than spend time and energy in planning. Successful projects allocate a significant amount of time to developing plans, documenting needs, specifying deliverables, developing data models, and reviewing business procedures that will be impacted. The planning process alone may take 20-25% of the entire project budget. Short-cuts in the initial stages often lead to delays and revisions in the future — potentially de-railing the entire project if major aspects are overlooked in the early stages.

Manage expectations.

The executive sponsor must be sensitive to the challenges every project faces and the human tendency to minimise change. Users often want to force a new system to fit their old office procedures, rather than change the way they do business. The IS folks may hold firm in their commitment to technology standards, completing disregarding the user's desires.

The executive sponsor must be willing to watch for signs of resistance and deal with them promptly, before they turn into unconscious project sabotage. Human beings resist changes that are forced upon them. New technology systems, corporate re-engineering, massive down-sizing and general society pressures make it even harder for people to deal with the day-to-day pressures of corporate life. Understanding the diverse humanistic forces at work require different skills than building a new widget or building a new system.

A natural tendency to make a project "better" as it develops can lead to *Scope Creep* (an ever-expanding project scope). As the new system grows and becomes available for testing, many people will ask for *little* improvements. If the system has been well designed in the early planning and specification stages, these changes should be minimal. If users have very different expectations than the system can deliver, disillusionment sets in quickly. For example, in today's *Windows*-oriented world where people have computers at home or see exciting examples in the media, installing an old-style character-based system could lead to mass revolt in the office. Even if the IS department has not promised a fancy new system, there may be a general level of expectation on the part of users. IS pays the price of failure; the organisation pays the price of widespread dissatisfaction and low productivity. These issues must be identified *before* the project begins, not after it is rolled out.

Many organisations are using *prototyping* as a way of developing specifications and managing user expectations. For example, a model screen *(prototype)* is developed so users can actually *see* how a screen might look. This way, expectations and usage patterns can be uncovered very early. Then, the system can be built behind the scenes. With early involvement on a small scale, users have a sense of what is coming and have more time to adapt; IS folks know the user interface is acceptable early on. There

are fewer surprises down the road. Again, advance preparation saves pain all around.

Regular steering committee meetings.

Strong teams develop through regular contact and shared objectives. Regular meetings of the Project Steering Committee (Technology Council, or whatever it is called in your organisation) provide a vehicle for problem resolution and status updates. In long-term projects, the meetings may be weekly in the beginning (when specifications are being developed), then move to monthly (while the technology is being developed), then more to weekly again (as people prepare for the final stages).

A formal agenda and regular meeting time helps people stay on topic and move the meetings along efficiently. The chair of the committee must be committed to the overall process of guiding the meetings so they are effective, yet stay on topic. As people become more familiar with each other, meetings can degenerate to bitch sessions or to off-topic discussions. Each person's views need to be respected. However, sometimes off-line meetings need to address those concerns in order to keep the project on course.

Managing steering committee meetings effectively includes individual discussions between the Executive Sponsor, IS management and individual project managers in planning the meeting agenda. The results of these individual discussions guide the steering committee meetings. Walking into a high-level meeting and being hit unexpectedly with a number of critical issues is not a good way to build rapport.

Hold people responsible and accountable.

Responsibility includes: delivering on plans and commitments, being part of the team, acknowledging when there are problems and being willing to adjust personal priorities in favour of the overall project priorities.

Each member of the team must understand the rules of the project. And, they must understand the penalties for not following the rules. Too often, threats are made that have no force behind them (*Management by Intimidation)*, so people learn that the threats have no value. Most people work best when they know where they stand. When they understand what the rules are, what the rewards are and what the penalties are for not following the rules they are more likely to act consistently. In organisations where "rules" are based on personal agenda, ego, or unpredictable actions, motivation and productivity are impossible to maintain.

The entire team — starting with the Executive Sponsor and the Project Manager — are responsible for clearly defining what is expected from each member of the team. The team is also responsible for maintaining consistency within the team. If someone needs help, they are responsible for asking for help. If someone sees that another needs help, they are responsible for trying to assist. If someone sees that work is being done badly or improperly, they are responsible for reporting the problem to someone who can deal with it.

Occasionally, a single person does not fit in a project team. While every effort should be made to help them participate, sometimes the person must be re-assigned to a more appropriate job or fired from the project. Compassion and understanding cannot overcome incompetence. The sooner the reasons for someone's lack of performance are identified and dealt with, the sooner the entire team can return to productivity. Ignoring a person who is not contributing or who weakens the entire team effort is not acceptable; the entire team cannot be placed at risk because of one person's inability to perform, for whatever reason.

Certain things must be designated as intolerable: any abusive treatment or language, any discriminatory or derogatory words or actions, violence (including verbal assault) or any other physically inappropriate

behaviour. Should intolerable actions or language occur, management must deal with the person immediately, removing them from the team.

Do not micro-manage the project.

Provide leadership, not micro-management to the team. If you have hired the best project manager and developed a good rapport with them, you should trust them to handle the day-to-day tasks. Ask them to report on how well they are performing against your long-term goals and expect them to ask for your assistance on strategic and political issues.

Know when to step in.

If you have developed a good working relationship with the project manager and the team, and you get regular feedback on progress, you should not step in. If, however, there are differences between your strategic goals and the direction you see the project moving, it is time to pull in the reins a little.

Meet with the project manager and review your goals and expectations again. Make sure they are able to repeat back to you what they "think" they heard you say. Explain your concerns and your suggestions for getting the project back on track. Be willing to work together to find a better way. If you are faced with political pressure and the project becomes delayed, make sure the project manager understands the implications to them personally, as well as the implications for the entire project.

Know when to ask for help.

Occasionally, the dynamics or communications between a sponsor and the project manager (or the team) create a situation that is not working as well as it should. That is the time to ask for help — from other executives who have encountered similar situations, from the Human Resources department, or from trusted advisors.

In all the personality research done in the last few years, the prevalent (or "typical") personality type of IS folks is often very different from the prevalent personality type of executives and business-oriented people. This difference can create significant personality conflicts as well as communication difficulties. To resolve these differences, some training and/or individual consulting may be required to come to a mutual understanding of goals.

Writing off conflicts (or differences in perspective) as simply "personality conflicts" does not get to the root of the problem. Ignoring the problem will only make it grow worse. Dealing with it as quickly and as thoroughly as possible is the best solution. If it is not dealt with, the success of the project is at serious risk.

Do not accept guesses.

People naturally want to please their "superiors." This desire to please can lead to overly optimistic project forecasts. Lack of experience in similar projects also leads to underestimating implementation time. This where having the best project manager, the most experienced team members and a good schedule/task management system pays huge dividends.

The best project control tools provide for both capture of tasks completed against expectations, and a projection of the success rate for the balance of the project. For example, the initial project plan contains a list of tasks and the expected time for each task. Over time as tasks are completed, the expected time is compared against the actual time. If tasks tend to take 10% longer than planned, the remaining task estimates are automatically increased by 10% to reflect a more realistic effort. If tasks are taking 50% longer than expected the future tasks should be increased by 50% as well.

Be aware that industry research shows IS projects frequently taking 200% or 300% longer than expected; the failure rate for

projects *never* implemented is alarmingly high (31%). Do not be lulled into thinking your organisation is differently unless you have facts to support a better success rate.

Identify and solve small problems as they surface.

One of the reasons projects take longer than expected is that small problems appear constantly — and are ignored. People change their mind as they begin to see the results of the new system. They "thought" they wanted a report one way, then realise they need something differently — leading to changes in task times and dates. People assigned to the project change — and the requirements change as a result — leading to more delays. As inexperienced users become more educated, they find that their procedures must change; that can lead to system requirements changes — and more delay.

Time spent in the beginning of a project in system design, business process review and prototyping saves more than its weight in gold later on.

Prototyping sample screens and sample reports so people can "touch and feel" the end product before the system is created can unearth many of small issues. When people see what they will get, they begin to get excited about the final result. It becomes more real for them, rather than a system that will be imposed on them "some day." And, they have something tangible they can begin to work with in changing their business operations. Enthusiasm on the part of users is a major hurdle to overcome — the earlier it starts, the better the project will be.

IS people often fail to do the "people" parts of the system, feeling that their only job is to build good computer system. In reality, the system will only be successful if people are willing to use it. As the executive sponsor, your job is make sure that the people issues and solutions are as important as the technical issues and solutions.

Understand the dynamics of your organisation's culture.

Each organisation has its own unique corporate culture (or corporate personality). Understanding the strengths and weaknesses of your organisation is your job. Translating that culture into guidance for the project manager and team is also your job. Using the knowledge of the culture to create a successful project is both your job and the job of the project manager. You must both be "reading from the same script and singing the same song."

Projects often fail simply because the technical team failed to understand the corporate culture, or the executive sponsor did not guide the team through the land mines that exist in most organisations. For example, the project steering committee (or IT Senior Governance committee) expects regular reports on the progress of large projects. If a technical person makes a highly technical presentation, the audience may not understand or appreciate what is being said. You — as the executive sponsor — must make most high level presentations on the project's status. Or, you must groom someone to make the presentation in a way that can be understood by non-technical people.

Use your political clout, if necessary.

There will come a time when you must step in and get something done politically that the team cannot accomplish. Since your power must be used sparingly, be sure of the situation and all the facts before dropping the hammer on someone politically. For example, a business section manager is resisting the process of working with the team or is not making changes in their office to prepare for a new system. You could create havoc with them directly or their line executive (using a "big hammer"), or you could draw the executive into a conversation with the goal of helping everyone be successful (using the "velvet glove on an iron hand" approach). Ask for ideas about

how to break through the road-blocks. Let them handle the problem of their departmental resistance and report back to you.

Involving other executives in *solving* a problem goes a lot further toward to enhancing your own political clout than having a lot of people think you're a cold-hearted SOB! If you are the executive being approached due to resistance in one of your people, work with those involved to create feasible solutions rather than creating a new turf battle. The political benefits of a teamwork approach extend far beyond today's issues to building a reputation as being very strong, yet fair and reasonable.

Be the project's champion.

One of your most important roles is to champion the project. Talk about it to others and express excitement and enthusiasm about it. If you have problems, work with the appropriate people to resolve them. Do not bad-mouth the project or the people or you will look like the loser.

Remember Tom Sawyer and his enthusiasm about painting the fence. When he complained, no one wanted to stay around; when got excited about paining, everyone wanted to join in. People everywhere react in similar ways: the martyrs are left alone and the Pied Pipers draw a crowd. Be the Pied Piper, not the martyr.

Reward success.

Positive actions must be acknowledged and rewarded in appropriate ways. Small, consistent actions mean a lot: a "good morning" smile, "thank you" for a job well done, needed supplies always being available and a general level of respect for team members can do far more to keep the team motivated than a party once a year in the grand ballroom. Certainly, milestone celebrations with fancy speeches are nice. However, many people would rather have a day off, an engraved momento or an award they can frame. Others would prefer money or something they can share with their family.

Set a good example.

In everything you do, *set a good example*. Show people how well you can work with them by demonstrating your teamwork skills. Express enthusiasm by showing others how to stay excited and upbeat — even when problems surface. Tell others about the project's success by expressing excellent communications skills.

Constantly imagine yourself as the Executive Sponsor of the most successful project your organisation has ever had! This is the method used by the most successful business people and athletes — using your powerful vision of what is possible — to seeing it become reality. Act as if you are a leader and other people will sense it. Act as if you are the most successful Executive Sponsor ever and it will happen. This is not an ego trip. It is leadership at its finest.

Congratulations on a successful project!

Copyright 1995, Barbara Taylor.

This paper was first published in the January 1997 of the old (paper) version of Methods & Tools

Training and Process Quality

Franco Martinig, franco@martinig.ch, Martinig & Associates, www.martinig.ch

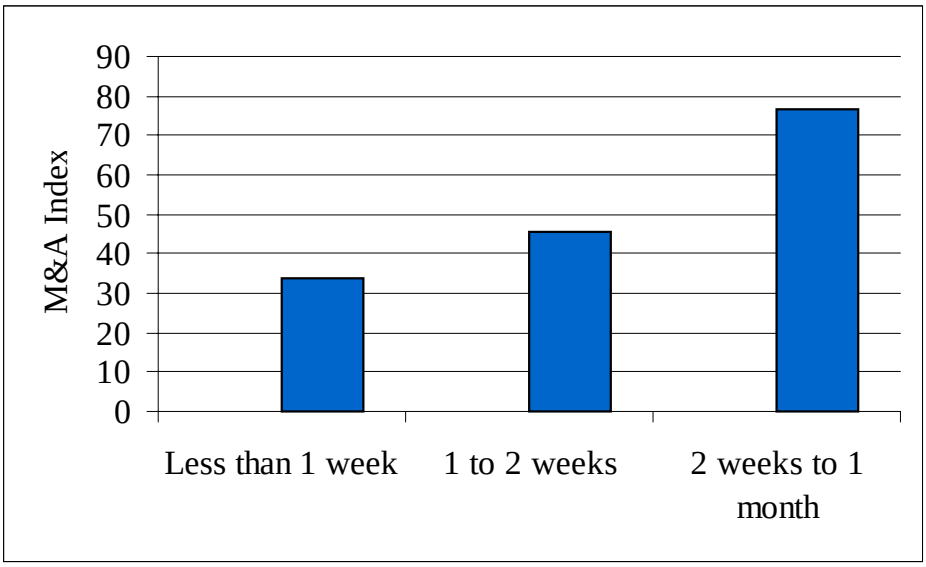
In our changing industry, training is needed to keep pace with the release of new tools and techniques, to get the right skills to handle promotions in the project management track or simply to get the most out of the new release of a software development environment.

Sadly, training can often be considered as the "lost benefit" of the software development jobs. You will often hear a lot about it in the offers or the interviews, but you will have few occasions to enjoy it in regular jobs... Bad luck, as it seems that organisations that offer more training to their software development employees enjoy a higher quality for their software development process.

This affirmation is based on software process evaluations performed in recent years with 63 medium to large software development units, mainly in Europe and North America. The Capability Maturity Model (CMM) of the Software Engineering Institute (SEI) is used as a basis to evaluate the quality of the software development process. (go to www.sei.cmu.edu for more information about the SEI and the CMM).

The quality of the software process is measured by the M&A Index, an aggregate value defined by Martinig & Associates that tracks implementation of good practices across all the maturity levels of the CMM. A high score for the M&A Index reflect the good quality of the software development process of the software development unit. In our evaluations, process-related items were completed with questions on technologies, staffing and project achievements.

Among the results of our evaluations, we found that one organisation out of three gives its software developers less than one week of training per year. One organisation out of two gives its software project managers less than one week of training per year.



Software engineering training and process quality

As we see on the first chart, organisations giving less than 1 week of training each year to their software developers achieve a quality level that is only half of the level achieved by organisations that give their software developer more than two weeks of training. A similar difference, a little bit smaller however, can be detected in the second chart comparing quality levels and the amount of project management training.

As in the chicken and egg story, it is difficult to determine if the aim for process quality determine the level of training or if the level of training is a contributor to the quality of the software development process. We know however that most of the organisations surveyed are not explicitly quality oriented. Only few of them have implemented quality programs like the ISO 9000 series or the CMM. Taking the opposite approach, it seems clear that companies that give little training to their software development employees have a lower quality of the software development process.

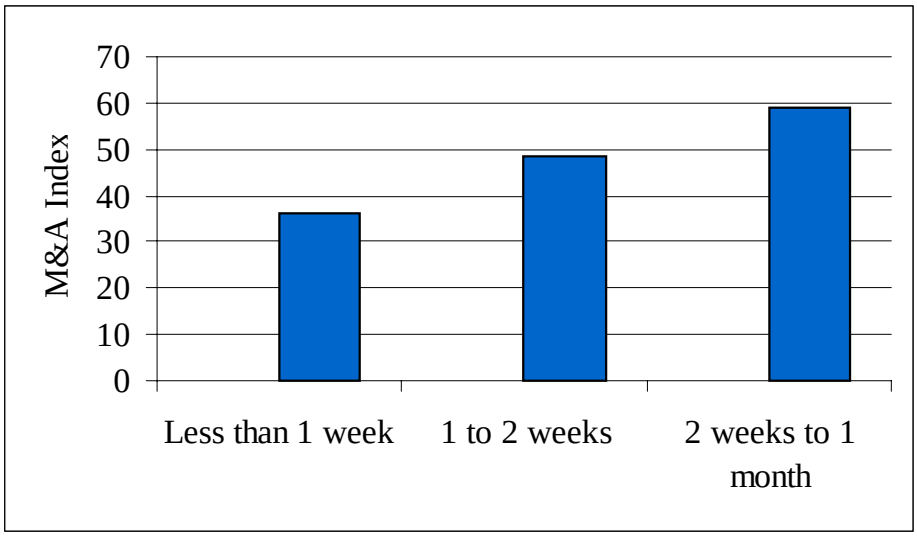
We can therefore say that it is not possible to have a high quality software development process without giving the appropriate level of training to software development employees.

Of course, we know that the quality of the software development process is in most corporation at best a nice to have in some slide show about the IT department. But our data shows that the process quality measured by the M&A index is also linked to the respect of estimated deadlines and budgets for software development projects. (more about this in our next issue) This is concrete information that sounds pleasantly to the ears of the supporters of software process improvement and should motivate every software development unit to invest in a decent training program.

Martinig & Associates perform free process evaluations based on the Capability Maturity Model. You can download the questionnaire and see some global results on our web site www.martinig.ch/ae/procase.html.

The confidential individual evaluation results are sent to participants as a PDF file.

A special report concerning evaluations in North American and European banks is also available www.martinig/ae/bk98.html



Project management training and process quality

Products

New JBuilder from Inprise

Inprise/Borland (www.inprise.com) has just released at the beginning of September the new version of its Java development tool. The tool supports the latest Java version and offers functions for remote debugging.

Besides the technical news, this event is important as a sign of the return of Inprise/Borland to the "serious" activity of producing tools for the software developer after its failed merger with Corel. The "back to the root" feeling is reinforced by the fact that the old Borland name is making a strong comeback in the company's communication items. Without judging the products' qualities, it is in the best interest of the software development community to have another strong competitor in our industry.

Visible releases Analyst 7.5

Visible Systems (www.visible.com) has released at the end of August a new version of its Visible Analyst. The main new feature is the support of the UML diagrams which ads to the feature of XML generation.

This release is another example showing that the modelling war has been definitively won by the UML, after its adoption as a standard by the OMG. There is still some competition left at the methodology level, but there are many chances that the Rational Unified Process will also dominate the commercial software development world, even if some other organisations, like the OPEN or DSDM consortiums, are offering alternative approaches.

Companies

Hot Summer at the Head

It has been a hot summertime at the top of some of the largest software companies with three CEOs saying good-bye to their job.

At Corel, the chief executive and founder Michael Cowpland has stepped down from the top spot. After failing its marriage with Inprise/Borland and making a huge bet on a desktop Linux market that has not (yet?) materialised, Corel found itself in very difficult financial position. About 20% of the work force (320 jobs) have been cut as a direct result of the failed merger. Just at when closing this edition, Corel is getting a large cash infusion from Microsoft!

Informix announced also that it will cut about 500 jobs as part of an effort to save 70 to 80 millions of dollars a year. This announcement was made after second quarter revenue fell 4% from the same last-year period. CEO Jean-Yves Dexmier was ousted only one year after its nomination. The new

Improve Your Software Development!

Subscribe to *E-ssentials!*, a **FREE** biweekly e-newsletter with how-to's for successful project management and software development. Our tips, hot topics and techniques from industry experts will help you conquer your everyday challenges!

"*In almost every E-ssentials! there is an article that makes me think about how we work and how to make it better. You have given good thought to the articles, the length, depth, and how often you send it.*" – Richard Schoultz, Solutions Architect, Cross Communication AB

Subscribe FREE at http://www.spc.ca/esse

www.spc.ca

CEO is Peter Gyenes, previously CEO of Ardent, a datawarehouse company acquired by Informix at the beginning of this year. In a reverse-take-over way, many former senior managers of Ardent have now been placed at the top of Informix.

Finally, Charles Wang, the CEO of Computer Associates (CA) has left its job, letting the operational direction to its long time number two Sanjay Kumar. CA announced it will refocus its business in three areas: security, network management and application development. Special divisions like the iCan-ASP will be spin off. Like its competitors BMC or Compuware, CA has recently suffered from the decline in sales in the mainframe area.

The common fact between these three situations is that the sharp decline of the stock price following the bad news announcement seems to be the most important catalyst for change than the customers needs. We will not say that this is a surprise, especially for CA...

$9%£ Numbers

R.I.P.

- Over 60% of all Web-access data resides on a mainframe
- Cobol mainframes process more than 83% of all transactions worldwide
- Over 95% of finance-insurance data is processed with Cobol

Source: "In Cobol's Defense", Roundtable, IEEE Software, March/April 2000.

I am not afraid personally of the death of Cobol and the end of the mainframe already announced in the 90s... but I will be much worried in 20 years after the death of the last remaining mainframe Cobol programmer!

Project Success (and Failure!) Profiles

Here are some numbers from the famous Standish Group research on how software development projects reach success, or more precisely failure. These numbers are the result of a study conducted with 365 organisations representing 8'380 applications. According to the study, 31% of the projects were cancelled before ever getting completed.

Project Success Factors

1. User Involvement	15,9%
2. Executive Management Support	13,9%
3. Clear Requirements	13,0%
4. Proper Planning	9,6%
5. Realistic Expectations	8,2%
6. Smaller Milestones	7,7%
7. Competent Staff	7,2%
8. Ownership	5,3%
9. Clear Vision & Objectives	2,9%
10. Hard-Working Staff	2,4%
Other	13,9%

Project Challenged Factors

1. Lack of User Input	12,8%
2. Incomplete Requirements	12,3%
3. Changing Requirements	11,8%
4. Lack of Executive Support	7,5%
5. Technology Incompetence	7,0%
6. Lack of Resources	6,4%
7. Unrealistic Expectations	5,9%
8. Unclear Objectives	5,3%
9. Unrealistic Time Frames	4,3%
10. New Technology	3,7%
Other	23,0%

Project Impaired Factors

1.	Incomplete Requirements	13,1%
2.	Lack of User Involvement	12,4%
3.	Lack of Resources	10,6%
4.	Unrealistic Expectations	9,9%
5.	Lack of Executive Support	9,3%
6.	Changing Requirements	8,7%
7.	Lack of Planning	8,1%
8.	Didn't Need It Any Longer	7,5%
9.	Lack of IT management	6,2%
10.	Technology Illiteracy	4,3%
Other		9,9%

Source: Standish Group, Chaos Project, http://standishgroup.com/visitor/chaos.htm

The evidence is here: users are still the biggest problem in software development projects...:-]

In Others' Words

Is the Web different?

"About every 10 years or so, a major new software related technology captures the industry's consciousness. Avant garde software folks claim it as their own, and in so doing, become the darlings of the technological scene. High salaries, considerable prestige, and no small amount of hubris are sure to follow. The corps of avant garde adherents argue that the new technology is truly different, requiring a new "paradigm." The ways of the past simply don't apply. In fact, the old ways can't possibly be adapted to a new set of business rules and technological realities. As a result, the avant garde reject the disciplines of the generation that preceded them, but ironically, adopt approaches that failed miserably a few generations back.

The Internet and the vast array of applications that it has spawned are undoubtedly a major new software related technology. I won't bore you with the obvious clichés; suffice it to say that the Internet and the WebApps that populate it are big - very big - and that their impact is profound.

What worries me is that this major new technology has become a breeding ground for important WebApps that are hacked in much the same way as important application software was hacked a few generations back - in the 1960s and 1970s.

WebApps have to be hacked, argue the avant garde (who, of course, would never use the word "hacked"), because:

- WebApps must be developed in days or weeks - time frames that don't allow for anything but a rush to the finish line.
- WebApps are constantly evolving - so why spend time specifying what's needed and designing how to build it when everything will change anyway?
- WebApps are inherently different than application software - content (text, graphics, images, audio, and video, for example) is inextricably integrated with procedural processing.
- The people who use WebApps are more tolerant of errors. What users really want are cool Web sites that are up and running in days, and besides, it's almost impossible to know what WebApp users really want, because the demographics of Web visitors are so hard to predict.
- The people who build WebApps are different - free-thinkers all - who certainly would feel unduly constrained by the old ways. In fact, talk of disciplined approach - other than build it, test it to death (if time permits), and then put it online - usually results in grimaces all around.

Of course, I'm overstating my point a bit, but you get the picture. Too many WebApp developers make these statements in an

attempt to erect barricades against an old school view that suggests a disciplined, engineering approach to the creation of business critical Web-based systems.

[...] This leads us to the pivotal question: Can software engineering principles, concepts, and methods be applied to WebApp development? I believe that many of them can, but their application may require a somewhat different spin.

[...] I contend that software engineering principles always work. It's never inappropriate to stress solid problem solving, good design, and thorough testing (not to mention the control of change, an emphasis on quality, yadda, yadda). A specific software process might fail because it is overkill, or the work products it requires are unnecessary or burdensome, or a person or team becomes overly dogmatic in the application of the process. But history is on the side of a solid engineering approach.

[...] There is much managers can learn from the new generation of WebApp developers - their enthusiasm, creativity, technical competence, and innate understanding of what makes a good WebApp must not be ignored. But there are also things that the new generation can learn from those of us who have been around the block a few times. The question, really, is whether any meaningful learning will occur (in either direction).

[...] The philosopher George Santayana thought hard when he made the comment: "Those who forget the past are doomed to repeat it." But maybe that's our karma in the software biz. There's absolutely nothing that I've seen over the past 30 years to make me think otherwise."

Source: "What a Tangled Web We Weave", Roger S. Pressman, IEEE Software, January/February 2000

Do you want to change your karma?

Coming next in Methods & Tools

- Software Configuration for the Web
- QA for the Web Project
- Data Warehouse Design

Classified Advertisement

Advertising for a new Web development tool? Looking to recruit software developers? Promoting a conference or a book? Organising software development related training? This space is waiting for you at the price of US $ 20 each line. Reach more than 7'500 web-savvy software developers and project managers worldwide with a classified advertisement in Methods & Tools. Without counting those that download the issue without being registered! To advertise in this section or to place a page ad simply send an e-mail to franco@martinig.ch

METHODS & TOOLS (ISSN 1023-4918) is published by **Martinig & Associates,**
Rue des Marronniers 25, CH-1800 Vevey, Switzerland Tel. +41 21 922 13 00 Fax +41 21 921 23 53
Editor: Franco Martinig; e-mail franco@martinig.ch
Free subscription: www.martinig.ch/mt/index.html
The content of this publication cannot be reproduced without prior written consent of the publisher
Copyright © 2000, Martinig & Associates