

## White Paper

### SAP User Interfaces – Insights from practical experience

By Brent Talbot

In a previous white paper “SAP user Interfaces” I challenged SAP implementers to look more closely at the tools available to create better user experiences for SAP users. Over the past year I have been challenging myself and my project colleagues to use the tools. This has led to some interesting results.

In this paper I have provided details of some of my experiences using the tools for creating alternative screens for SAP users. I have kept the details of each project brief only providing the essential details. In this way I have covered many different options without writing a long dissertation.

The remainder of this paper discusses the content provisioning options but before discussing these options I would like to briefly mention the rendering engines – Portal and NetWeaver Business Client (NWBC). Both of these SAP rendering options provide a way to present different content using role based menus in a unified user interface. In this was the developer can use different content provisioning tools and present the results in a way that the user is unaware of the technology used to provide the content. All of the content provisioning options discussed can be display in either Portal or NWBC. The difference between the two rendering tools is beyond the scope of this paper but at a very high level Portal is browser-based with a zero client install, while NWBC provides a rich desktop client.

In sharing these experiences, I hope to provide some practical insights into the different options for creating user interfaces for SAP users conclude the paper with a summary of what I believe are the lessons learned from these experiences.

#### Web Dynpro Java

One of our clients was engaged in a significant roll out of SAP into its retail stores but the functionality required by the store users was reasonably limited. It was an ideal candidate for creating new interfaces.

The customer made a good start by installing the SAP portal and enabling single sign on. The portal was also branded, which gave the users a sense of ownership of the system. The next step was to create some new screens to improve the user experience for the common transactions the store would be using. The choice was made to use Web Dynpro Java (WDJ).

Because this was a new area for the customer and they were unsure about what could be achieved or what their retail staff would be happy to use, the program specifications were short on detail. Developments were

#### About the author

Brent Talbot has over 20 years experience in IT, spending the last 13 working with SAP.

Brent's focus has been on SAP development and he has practical experience with many of the SAP development tools.

Brent is currently the Technology Team lead at Soltius New Zealand.



completed, shown to the users and then adapted further.

In this environment, the WDJ development required three developers to be involved in any one enhancement:

1. An architectural role to design technically how the WDJ application would interface with the SAP business layer.
2. A person to build the Web Dynpro Java (WDJ) application
3. A person to adapt the SAP business logic to expose it to the WDJ environment.

*“For this customer site, WDA was up to four times more efficient than WDJ.”*

Some very good screens were developed using this approach but it took too long and cost too much to develop. A lot of time was also spent redeveloping search helps and data validation which occur automatically in the SAP back end system.

The other point of pain in the WDJ environment was in migrating changes. The Java developments were controlled by NWDI (NetWeaver development infrastructure), and the ABAP developments by the usual SAP transport system. The WDJ developments are client based and the ABAP developments server based. While the customer managed this complex environment well it required additional administrative work to set up the environments and resolve issues.

My involvement led to the first use of Web Dynpro ABAP (WDA) as an alternative. The approach was different because only one developer was involved, working closely with the business analysts and the store users to create and adapt the entire application. The reuse of existing SAP objects also reduced the development time. All the changes were controlled by the transport system already in place with no administrative overhead.

For this customer site, WDA was up to four times more efficient than WDJ. I was then asked to train the in-house ABAP developers in WDA, delivering the standard SAP course, and concluded it by setting one developer the task of creating a real WDA application. I returned a week later to answer questions and help resolve any issues. There were a few issues to resolve but these were easily sorted during the day and the application was completed. The customer has now standardised on WDA as a development tool.

## Conclusion

- WDA is more suited to a rapid development environment than WDJ for SAP centric applications because of the ability to reuse existing SAP objects.
- The ability to use one developer to complete the entire application reduced the development time and cost significantly.
- One developer working closely with the business analyst and end user allowed for better communication which resulted in screens more suited to the end user
- It is only a small transition for existing ABAP developers to learn WDA
- The integrated transport system with WDA reduced the administration overhead of moving changes between systems.

## Business server Pages (BSP)

CRM makes extensive use of Business Server Pages (BSP) as a way of creating browser based screens. The standard BSP tools have been enhanced by SAP to embrace the model view controller concept and also to generate a number of the objects.

For a recent project, we were given the task of customising some of the standard screens. This was easily achieved using the tools provided in CRM with very little development effort. The final screens were not significantly different from the screens provided by SAP, but by making small changes the users found the system easier to use. If the

screens had instead been re-developed in WDA, considerably more resources would have been consumed.

## Conclusion

- When considering changing standard SAP screens try to use the same tools that SAP have used
- BSP offers a viable alternative for creating browser based screens but to create new BSP screens will take longer than using WDA.

## Internet Application Components (IAC)

One of the older technologies that SAP has used to create browser based screens is Internet Application Components (IAC). I worked on a project utilising SAP Retail Store with IAC.

The customer wished to make changes in the goods receipt and stocktaking areas.

For goods receiving there were user exits available to achieve most of the changes the customer wanted. The screen layout change, however, required the use of HTML and HTMLB coding. The stocktaking changes did not have readily available user exits.

The first approach for the stock taking changes was to use the enhancement framework to create implicit enhancement points, and to make changes to the HTML code. This proved to be too cumbersome and after reviewing the amount of work required I replaced the screens with WDA. The business logic was already exposed by the coding behind the IAC screens, making the creation of the WDA screens easier.

It should be noted that SAP is providing a WDA based solution for SAP Retail Store in Enhancement Pack 4.

## Conclusion

- Small changes or changes supported by configuration and user exits are most effectively done with the same technology used by SAP – in this case IAC.
- More significant changes are best done by replacing the existing screens with WDA.

## Solution Manager Test WorkBench

I was asked to assist a customer with deploying Test Workbench as part of a solution manager implementation. Test workbench offers two different views, one in the SAP GUI and the other using WDA screens embedded into the SAP GUI.

There was some debate about which of the two screen choices the users were to be trained in. Testers made up the majority of the user base. They would be involved in executing the test scripts and updating the status of the tests. The Web Dynpro screens were chosen because it was felt they were more intuitive and easier to use.

There was more debate, however, over the test administrator screen. This is used for creating test plans and packages, allocating work to users and reporting on progress. It was concluded that people performing these functions would be allowed to use either of the screen options, with some even choosing to use both.

The other users involved with creating the test cases and setting up the project structure did not have the choice of using WDA and were only provided with SAP GUI screens.

## Conclusion

- Browser-based screens are best suited to users who will be using transactions with limited functionality.
- Power users or those with more complex tasks may prefer to use the SAP GUI transactions.

## Standard WDA Business Package

SAP has delivered standard business packages using WDA as an alternative to the SAP GUI transaction. I have been

involved with two customers in delivering the maintenance technician business package.

The standard WDA screens delivered and the accompanying roles are very comprehensive and like some of the SAP GUI screens contain a lot of data fields and buttons. The screens can be configured and in the projects I had been involved in the functional consultant had done a good job but the screens were still too cluttered for the users.

This is not a criticism in any way of SAP developers or architects, who I think in general do a tremendous job. The reality is the screens SAP delivers in the standard package will always be generic and will not precisely meet customers' requirements. The question then arises – how much money is the customer prepared to spend to change the look and feel?

*“For this customer site, WDA was up to four times more efficient than WDJ.”*

I was asked to look at the screens to see what more could be done. WDA provides standard tools which can be used to change the look of a screen without changing the underlying application. At the simplest level, this can just be hiding screen elements but it can go further to allow for changes in the other attributes of screen elements, such as the caption. We did not have NetWeaver enhancement pack 1 on this project, which in addition to the other configuration options allows for more comprehensive changes using the enhancement framework.

The standard screens were altered using the configuration options and the customer was happy with the result.

## Conclusion

- SAP is delivering standard content using WDA
- The adaptation and configuration techniques available in WDA allow the standard content to be changed to more closely meet the user requirements without making any programming changes
- The configuration options are easy to use and do not require programming skills.

## WDA for Displaying Data

The customer call centre was using an excel spreadsheet to respond to customer enquiries about pricing. The spreadsheet was large, took a long time to load, was cumbersome to update and the call centre staff had to navigate through the spreadsheet for each customer enquiry.

The customer wanted to build a web page to display the pricing information and allow the call centre staff to be able to enter simple queries to reduce the amount of information displayed.

The business already used SAP and had in house ABAP skills but no experience in publishing SAP data on a web page. SAP had a reputation in the organisation of being difficult to use and slow to change. This was the reason in part for developing an Excel spreadsheet in the first place.

The user requirements were not well defined, but rather were based around having a better spreadsheet. After discussion with the business analyst we built a prototype in WDA and showed it to the users without telling them it was SAP.

The in house staff were familiar with the SAP data and so worked on extracting the data and provided custom function modules for retrieving the data. My role was to create the WDA to display the data.

In three days we had a working prototype which was capable of being demonstrated to the users. The development was no more than a selection panel and a table of data. The ability to use the normal SAP selection screen elements and search helps and the table component allowed this development to proceed very quickly.

The prototype resulted in a large number of requests for changes. At first I was concerned but after further questioning of the business people we established it was not that they did not like what they saw, rather they saw the possibilities and wanted to use the web interface more. When I explained that it was SAP, they were surprised

but also did not really care. It was a web page and was delivering what they needed and so the tools used to create it were of no concern to them.

It took another week to develop all the approved changes and the application went to production.

## Conclusion

- The users are interested in the end result not the technology used to get there
- WDA can be used as a rapid development tool
- Browser pages incorporating simple queries can be a powerful way to display SAP data for casual users.

## WDA for Workflow

SAP workflow has been used for many years. I developed my first SAP workflow in 1997 and one of the things that always annoyed me is the standard decision task with the buttons that stretch across the whole width of the screen.

In 2009 I had the opportunity to work with a customer on improving these screens. The customer was already using SAP Portal and users were familiar with the Universal Work List (UWL). This combination provided the opportunity to use WDA to replace the standard decision screen.

The first task was for purchase requisition approval. The users approving these requisitions were not regular SAP users and so to force them to use an SAP transaction they were unfamiliar with would not have received easy acceptance. There is also ready availability of BAPIs to update and release the purchase requisition. It was an ideal candidate for a WDA screen.

The WDA screen was developed in less than a week, providing a summary of the requisition using terms familiar to the customer and providing buttons for approval or rejection. The approvers readily accepted the application and it then lead to further work to enhance the other workflows.

## Conclusion

- WDA can be used to provide a better user experience for workflow decision steps.
- The development time for replacing the standard decision steps with WDA is short.

## SAP screens using .NET

Two Soltius customers have recently built screens using Microsoft's .Net environment for SAP data. One built relatively simple vendor enquiry screens to support purchasing activities by non SAP users. The other undertook an extensive replacement of SAP screens using a totally customised user interface.

For the first customer the webservice was created in SAP and the .NET application consumed it. The .NET developer did require assistance from a Soltius ABAP developer with .NET skills to make the process work but it was achieved relatively quickly and easily.

The second customer's project was more ambitious. There were many issues to resolve due to the stateless nature of the browser screens and the stateful nature of SAP transactions. There was friction between the .NET and the ABAP developers, as neither really understood the limitations of each of the environments. The project did deliver the customized screens requested but at considerable cost. The primary reasons for the high cost were:

- The inability to use the SAP delivered search helps and functions
- The technical design being completed without reference to the way standard SAP transactions operated. In effect SAP was treated as a database and not a business application.
- The use of multiple developers across different technologies.

## Conclusion

- The use of non SAP tools like .NET can lead to substantial cost to deliver browser based screens.
- The use of non SAP tools can lead to the inability to use standard SAP functionality and so increase the development costs.
- Small development in .NET can be efficiently done but larger developments should be carefully designed to ensure the underlying benefits of using the SAP packaged solution are not lost.

## SAP screens using .NET

I have been involved in the deployment of interactive PDF forms for a number of clients. An SAP HR project had a requirement for the approval of the payroll by various managers. The company had a large number of waged staff to pay each week. Collecting and approving the data had to be completed in a very short time frame.

The collection of the data was automated using Excel spreadsheets, which the contract managers were familiar with. The spreadsheets were automatically loaded into SAP and progress was tracked using standard SAP GUI screens.

The PDF forms were generated and emailed to the relevant managers for approving timesheets once the data had been received. The managers were provided with all the information they needed on the form, including variants and the ability to enter comments. Most importantly they could approve or reject the payroll. The completed forms were then emailed back into the SAP system for processing.

The use of PDF forms and email required little training for the managers and it was readily accepted by them.

It should also be noted that the development time was reduced considerably in this project because the customer had installed NetWeaver enhancement pack 1 which provides some useful technology improvements for receiving PDF forms and extracting the data.

## Conclusion

- PDF forms are readily accepted by users as a way to enter or review data
- SAP can efficiently produce and receive PDF forms as part of a business process
- PDF forms provide an easy way for users to be part of a business process when off line from SAP.

## Conclusion

In looking back at the projects in which I have been involved, it is clear that users are looking for a different user experience from the SAP GUI. It is also apparent that there are benefits to be gained from providing better user experiences.

There is no clear winner in the technology choices with BSP, WDA, IAC, PDF and .NET all having been successfully deployed. The choice comes down to looking at the starting point and the desired end point and then analysing the effort to achieve the end result.

For example, in CRM the starting point is most likely to be BSP. If no significant changes are required then the best choice will be to continue to use BSP. On the other hand, if the starting point is an SAP-delivered WDA screen, then using the customisation and adaptation tools available would most likely be the best option.

If I had to choose the technology I was most impressed by it would be WDA. Easy to use, particularly for existing ABAP programmers, WDA makes use of the existing SAP functionality, version control and transport system. My work with WDA this year has shown that it is the best tool for creating new web based screens for SAP data.

There is also a note of caution when looking at developing new user interfaces to SAP using tools such as WDJ and

.NET, which only interact with SAP through web services or remote function calls. In my experience these are the most expensive tools to deploy, not due to the merits of the technologies themselves, but because so much is lost from the SAP delivered functionality when only web services are used to communicate. The other factor is that there are likely to be additional technical people involved who do not have experience with the SAP framework, which may lead to conflicts in design philosophy.

I will continue to challenge SAP implementers to improve the user experience of SAP users. I believe the tools are available to achieve this in a cost effective way but it does require the people involved in the implementation to look carefully at the options available.

## Glossary

Term	Definition
.NET	Microsoft's web services architecture. Used for creating web pages on a windows platform. Interacts with SAP via remote function calls or web services.
BSP	Business Server Page. An SAP provided tool for creating web pages. The application comprises a set of pages that represent a user interface (layout and page flow definition). The layout part of a page normally contains ABAP or JavaScript code which is inserted into the HTML structure.
Enhancement Packs	SAP delivers enhancement packs for the SAP solutions from time to time to deliver new functionality without requiring a major upgrade.
IAC	Internet Application Component. A SAP provided technology which allows SAP GUI screens to be replaced by web screens. The application requires the use of HTML, HTMLB, ABAP and javascript
HTMLB	HTML-Business. A SAP provided extension to HTML providing some additional tags to allow the easy and consistent rendering of screen elements
NWDI	NetWeaver Developer Infrastructure. A SAP provided toolset for managing the java program code. The toolset includes the managing of software versions, checking in and out of program code and the movement of the changes through the system landscape.
PDF	Portable Document Format. PDF is a file format created by Adobe Systems 1993 for document exchange. It has been extended since then to allow additional functionality such as form filling. SAP NetWeaver 7 delivers an integrated Adobe Lifecycle solution which allows the creation of PDF documents from within the SAP framework.
WDA	Web Dynpro ABAP. A SAP provided tool for creating browser based screens. It is a model based tool which includes a layout manager and predefined screen elements. Program coding is completed in ABAP. Development is completed in the normal SAP development environment and all the SAP functionality is available directly to the WDA environment.
WDJ	Web Dynpro Java A SAP provided tool which uses the eclipse IDE for development. The development concepts are similar to WDA but coding is completed in java rather than ABAP. Communication with the SAP system is via remote function calls only.

### Legal Disclaimer

This paper is provided by Soltius (NZ) Limited to provide introductory assistance. It incorporates content from various web sites and vendor presentations including those provided by SAP. The views expressed in this document do not purport to represent those of SAP nor any other organisation. While reasonable care has been taken in authoring this document, Soltius accepts no liability or responsibility for any errors or omissions it might contain. This document's content should not be relied upon to make purchasing decisions.