TRANSALTA CORPORATION

Building a launch pad for native applications via your portal

How-To Guide

BUILDING A LAUNCH PAD FOR NATIVE APPLICATIONS VIA YOUR PORTAL

How-To Guide

110 12th Avenue SW • Box 900 Station 'M'
Phone 403.267.7632 • Fax 403.267.4705

Table of Contents

Tier 1 Applications

Before you start coding, you need to determine what are the common applications that all users require when launched via your Portal.

To build a launch pad or web page that contains a tool bar, much similar to the start menu in Windows you need to make some initial considerations, such as what applications should reside there. That is, what are the most commonly used applications by all users in your environment that should be included as the Tier 1 set of applications. When we refer to the term 'Tier 1', we mean, those applications that are also installed on every desktop and laptop throughout the organization.

The most likely candidates to make this decision are the entire Microsoft Office suite of applications, such as Word, Excel and PowerPoint. Access, depending on your organization's licensing scheme, may not exist on every desktop or laptop, so it is not always a suitable choice. Another set of worthy candidates would be Adobe Acrobat or the corporate standard for shared document viewing. In most organizations, the PDF format is widely used when distributing information throughout the organization, and when downloading content via the Internet. The last set of choices would include the corporate standard for e-Mail client, such as Outlook or Lotus Notes, the ERP client, such as SAP or PeopleSoft, and the ever popular file manager, such as Windows Explorer. There may be other tools, unique from one organization to another. The best approach is to work with your desktop infrastructure team and determine which applications are installed as part of a base image – and of course this may vary from one Operating System version to another, such as Windows NT 4.x and Windows XP Professional.

Once you've determined what your organization's Tier 1 applications are, you can proceed to determine how you will deliver your launch pad to end users, and whether there should be any restrictions, based on where they access your Portal.

Where a launch pad is appropriate

The biggest factor of where a launch pad is appropriate is how your web browser is configured as part of the corporate desktop or laptop standard.

You just can't deploy a portal launch pad for your native applications without investigating where it is an appropriate solution or where it can't even be delivered in your environment. A number of factors must be considered before making a final decision as to where you want your launch pad to be presented to end users.

The first and likely most important factor is how your web browser is configured for use on users' desktops and/or laptops. In virtually every organization, regardless of Operating System version, your Internet Explorer program is launched like any other native application – either via the start menu or an icon on your desktop. If that is the case, providing a launch pad for users may not necessarily be appropriate, especially if you are using Windows XP. In Windows XP, you now have the option to add quick link icons on your start menu taskbar. If you already have icons there, why would you want to deliver the icons in your portal? If you are using an older Operating System, then the launch pad is certainly a valuable option, especially if your portal is to eventually become your organization's desktop of choice – where users can get to all their applications and do all their work, and collaborate.

Another factor to consider is where it is possible to deliver a launch pad solution, especially if users are accessing their applications from outside the company firewall. One common example of this involves the use of a technology known as Citrix. Essentially, Citrix enables users to log in to their corporations network externally and access applications much similar to those available on their own workstations. The disadvantage is that the technology is so dynamic you can't easily configure your launch pad to work with it, without some heavy customization. If you choose to do the customization, make that decision now, otherwise make note of what are the fixed IP addresses of your Citrix servers, as you will need this information later in this How-To Guide.

There are also a number of other factors that may influence your decision to deliver a launch pad solution altogether in your organization via your portal, or Intranet. But, whichever option you choose, consider also the duration for which the launch pad approach would be valuable. Also, what is the foreseeable future of your portal or Intranet, and is it intended to be the desktop of choice, or simply a one-stop for a serious of common tasks, applications and collaborative tools. If the choice to proceed with building a launch pad is feasible at this point, then you can begin the construction process.

Building the launch pad

Now that you've made all the business decisions, it's time to design and develop your launch pad solution.

There is a common misconception that you have to create so many security holes in your web browser to make launching native applications possible. You may have to enable scripting from untrustworthy sources or turn off some degree of security checking to use code in ASP or JavaScript that can launch applications natively. Why bother with both the concern and the political issues this could create, when you could use a simple approach of building application URLs through settings in your desktop registry. Furthermore, your code is only as good as the standardization of the desktop image across your organization.

To elaborate, there is an object known as WScript that is available through both ASP and JavaScript (really JScript) for Internet Explorer. If you instantiate this object, you have the capability of performing either OLE (Object Linking and Embedding) functions against native applications, such as Microsoft Word, Excel and PowerPoint, or launch other applications through a *Shell* command, as if you ran the executable from the *Start-Run* method in Windows. Well, unless every application is installed in the same folder on every desktop across your organization, quite possibly this is a valid approach. But, if this isn't the case, which probably represents 99% of what occurs in organizations nowadays with non-standard desktops, then you have to consider every unique configuration. The help desk calls this is bound to create should be enough to drop this idea in a heart beat. And, in order to enable this you have to open up a bunch of security holes, that are not recommended by Microsoft, in Internet Explorer. There is a better approach, and it considers all the issues we've raised.

The approach that is recommended is to create application URLs, such as 'notes://' that invokes the application natively. Furthermore, you could specify a fully qualified web address to a document that the application supports so it is automatically loaded when the application URL is specified. So, for example, if users require access to the Lotus Notes address book, you could specify the following:

notes://calgary.transalta.ab.ca/names.nsf

Where 'calgary.transalta.ab.ca' is the name of the Lotus Notes domino server and 'names.nsf' is your Names & Address book on that server. What would happen is that your Lotus Notes client, if properly configured (details to follow), would launch and automatically load this database when the client is ran and you've

authenticated into the Notes environment. How is this accomplished ? By a simple set of registry settings, that's how. The registry entries you require establish the application URL such that it can be recognized by Internet Explorer. It's format (in a .REG file) would be as follows:

```
REGEDIT4

[HKEY_CLASSES_ROOT\{application}]
@="URL:{application} Protocol"
"URL Protocol"="URL:{application} Protocol"

[HKEY_CLASSES_ROOT\{application}\DefaultIcon]
@="{path to application executable}"

[HKEY_CLASSES_ROOT\{application}\shell]

[HKEY_CLASSES_ROOT\{application}\shell\open]
@="Open With {application}"

[HKEY_CLASSES_ROOT\{application}\shell\open\command]
@="{path to application executable} %1"
```

Here, the **{application}** represents the name of the actual application you wish to launch and use as part of the URL identifier or protocol when launching the application and/or launching the application with parameters (if they are supported). The **{path to application executable}** is simply the full path starting from the drive letter that specifies the program executable path. The only difference is that for the path separator symbol '\' you must specify two for each separator (ie. \\ instead of \ as the path separator). A working example of this for Lotus Notes would be as follows:

```
REGEDIT4

[HKEY_CLASSES_ROOT\notes]
@="URL:notes Protocol"
"URL Protocol"="URL:notes Protocol"

[HKEY_CLASSES_ROOT\notes\DefaultIcon]
@="c:\\program files\\notes\\notes.exe"

[HKEY_CLASSES_ROOT\notes\shell]

[HKEY_CLASSES_ROOT\notes\shell\open]
@="Open With Notes"

[HKEY_CLASSES_ROOT\notes\shell\open\command]
@="c:\\program files\\notes\\notes.exe %1"
```

One item to note is that, at the time of writing this document, not all applications work with the **%1** parameter, such as Microsoft Word, Excel and PowerPoint. As such you may not be able to support launching documents directly with all your Tier 1 applications. For the purpose of the launch pad, you won't need to worry about the parameter issue since you need only launch the application as if you were launching it from your Start menu in Windows. The next step is to create a .REG file that you can deploy to the entire organization via a tool like SMS, or whichever mechanism your organization employs to push registry changes out to networked workstations.

The next step to building your launch pad is to actually build the page that contains your launch pad. One approach, of many, is to simply build an HTML page containing icons, similar to the native applications on your desktop, that point to the associated application URLs you've created. A sample HTML snippet may look like the following:

```
<html>
<head>
<style>
body { overflow:auto }
</style>
</head>
<body leftmargin=0 topmargin=0 marginheight=0 marginwidth=0
bgcolor=#dddddd style="border-top:1px solid black">
<table border=0 height=100% width=100% style="border-
bottom:1px solid black" bgcolor=#dddddd>
<tr>
<td valign=middle align=left bgcolor=#dddddd>
<a href="tadoc://"><img src="doc.jpg" border=0 alt="Microsoft
Word"></a>
<a href="taxls://"><img src="xls.jpg" border=0 alt="Microsoft
Excel"></a>
<a  href="tappt://"><img src="ppt.jpg" border=0 alt="Microsoft
PowerPoint"></a>
<a href="tansf://"><img src="nsf.jpg" border=0 alt="Lotus
Notes"></a>
<a href="tapdf://"><img src="pdf.jpg" border=0 alt="Adobe
Acrobat"></a>
<a href="tasap://"><img src="sap.jpg" border=0 alt="SAP Logon
46d"></a>
<a href="tawin://"><img src="win.jpg" border=0 alt="Windows
Explorer"></a>
</td>
</tr>
</table>
</body>
</html>
```

Note that the applications that have been considered as Tier 1 in this example are (from left to right) are Microsoft Word, Microsoft Excel, Microsoft PowerPoint,

Lotus Notes, Adobe Acrobat, SAP Logon and Windows Explorer. The application url protocols that correspond to those applications are tadoc://, taxls://, tappt://, tansf://, tapdf://, tasap:// and tawin://, respectively.

Now, that you've created the launch pad page, you will need to determine where in the portal layout you wish to include it. The most popular choice for this would be the header, since it is always visible when navigating throughout your portal environment. Whichever choice you make think of convenience and how often users would access your launch pad.

To incorporate the new page into your header (or wherever you choose), locate the appropriate ASP page, such as the Header.asp or MyPage.asp or even the iiView.asp files that render the header, page and iViews, respectively. For the purpose of this document, we shall opt to place our launch pad in the header of our portal. Open a text editor and locate the Header.asp file, from the directory of your portal:

C:\Inetpub\wwwroot\SAPPortal\ie\portal

And, assuming you are using EP5 SP5 (or equivalent), under the table (after the </TABLE> tag), but before the </BODY> tag, enter the following:

```
<%
If
Instr(Cstr(Request.ServerVariables("REMOTE_ADDR")),"142.152.18
.") <= 0 Then
   Response.Write "<iframe id='taTaskbar'
src='/sapportal/ie/iview_runtime/com/transalta/desktop/stdapps
.htm' width=100% height=30 frameborder=0
scrolling=no></iframe>"
End If
%>
```

Make note that we have used a condition based on the IP address in this example, as we are considering Citrix in this code and as such the launch pad will not be visible when users access the portal through Citrix. Replace the condition with whatever IP addresses you wish to be excluded when displaying the launch pad. Also note the location of the launch pad page, in this example:

/sapportal/ie/ivew_runtime/com/transalta/desktop/stdapps.htm

Congratulations, you've now built a launch pad for your portal that should help drive your organizations vision of making your portal the desktop of choice.

Here is an example of what TransAlta's launch pad looks like in its current state:

This is still a work in progress, and based on a corporate branding initiative that is currently underway at TransAlta, the look and feel will likely change considerably before the final version is rolled out.

-Vis Naidu-
Systems Analyst
TransAlta Corporation
vnaidu@transalta.com