The National Archives

Government web archive: redirection technical guidance for government departments

v4.2

January 2010

This publication is available for download at nationalarchives.gov.uk.

Version History

1.0	First Release
2.0	Updated to include guidance on The National Archives custom handler for Internet Information Server (new section D)
3.0	Updated custom handler information and minor changes elsewhere
4.0	Updated to include changes to The National Archives custom handler, and additional guidance on Ionics Rewriter. Minor revisions elsewhere.
4.1	Slight change to Section 1 A, reinforcing that a 404 response should be served after an ‘archive checked’ redirection back from the web archive
4.2	Updated to include reference to latest version of Ionics (2.0) and changes to The National Archives bridging page used with the custom handler. Various minor revisions elsewhere.

Contents

Section A: Overview

1 Redirection procedure

This guidance covers three components that can be installed on government web servers to provide URL rewriting and redirection functionality, with specific reference to setting up redirection to the Government web archive where a requested URL does not exist on the department site.

This guidance focuses primarily on the technical aspects of setting up these components. One of these components – the Apache mod_rewrite module – ships with the Apache web server. The other two components are possible options for managing uniform resource locator (URL) rewriting and redirection on Microsoft Internet Information Services (IIS) 5.0 and 6.0.

Each component can be installed on departmental web servers and configured to handle unresolved requests. This can include issuing a redirection request to the Government Web Archive in order to attempt to retrieve the most recently archived version of the requested URL from there.

Figure1 shows the complete redirection process between departmental web servers and the Government Web Archive. The components are configurable in such a way that, whenever the requested URL is unresolved and matches a configured rule, the web server will respond to the requestor with a HTTP 301 status code redirection command to the rewritten URL. This process is seamless to the original requestor but the address bar in the user's browser window will indicate the redirected URL where a successful redirection has taken place. The components can also be used to implement URL rewriting to other locations on the same web server, in which case the originally requested URL remains visible in the browser, but the rewritten one is served to the user.

For redirection to the Government Web Archive, the flow of requests and redirections is described as below. However, note that this may not be the only redirection required, and the implementation may be required to support multiple redirections and rewritings depending on the

URL requested (for example departmental or website reorganisation may necessitate a redirection or rewriting to another URL on the same or another web server).

The redirection process for the web archive can be summarised as follows:

- User submits a request to department server
- **The request is resolved**
  The page or resource requested is found on the department server and returned to the requestor. No further processing is required
- **The request is unresolved and no redirection rules exist**
  No redirection rule is found so the department server returns a 404 error to the requestor. No further processing required.
- **The request is unresolved and one or more redirection rules exist**
  Department server is unable to resolve the request and the installed component catches this and parses the rules against the submitted request. A new URL is derived from the rules and returned to the requestor as a 301 redirect (assuming R=301 flag is set in Apache mod_rewrite or Ionics).
- The requesting browser automatically submits the new URL. When this is directed to the web archive server the format will be

  http://webarchive.nationalarchives.gov.uk/+/<original URL>
- **The request is resolved in the web archive (or other redirect destination)**
  The page or resource requested is found on the destination server and returned to the requestor. No further processing is required
- **The request is unresolved by the web archive**
  The archive server is unable to resolve the request. The web archive server issues a further 301 redirection to a page on the departmental web server that conforms to a specified pattern that won't trigger a standard 404 behaviour or circular redirection back to the web archive. The page ukgwacnf.html has been selected for this purpose.

  The departmental configuration must intercept requests for this resource and process them accordingly (by displaying an appropriate error page with a 404 response code). This can be achieved either directly via a page with this name, or preferably by using this as a dummy page and redirecting to the relevant error page.

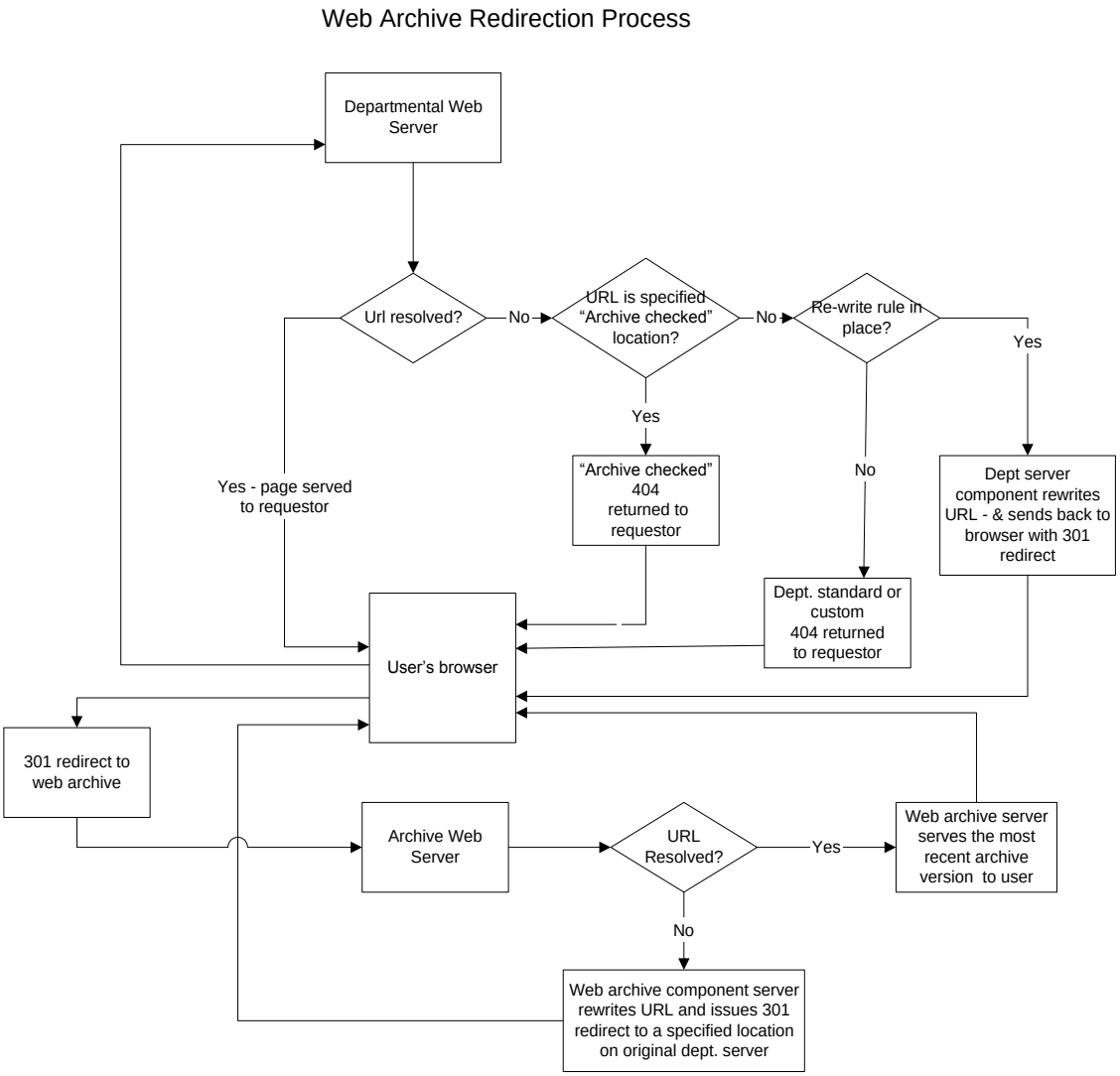

Figure 1: Redirection Process for the Government Web Archive

2 Supported Environments

The Web Server implementations to be supported are listed below. This list has been derived by an analysis of the most prominent implementations in use at the end of 2007.

- Microsoft Internet Information Server (IIS), Versions 5.0 and 6.0
- Apache versions 1.3, 2.0 and 2.2

Section B: Apache ISAPI Rewriter – mod_rewrite

1 Installation of the component

The mod_rewrite component is an open source component that can be used for URL rewriting and redirection on Apache web server installations. You can download it from the Apache website, such as http://httpd.apache.org/docs/1.3/mod/mod_rewrite.html for version 1.3. This site also provides links into further Apache versions, documentation and discussion forums.

As standard, Apache comes packaged with the mod_rewrite module installed but not enabled.

To check that this is the case with your installation use the following command to view a list of all compiled modules:

```
sudo /usr/local/apache/bin/httpd -l
```

If the mod_rewrite module is not present, you will need to build it from source using the commands outlined below:

```
./configure --prefix=/usr/local/apache --enable-module=so --enable-module=rewrite
make
sudo make install
sudo /usr/local/apache/bin/apachectl start
```

To enable the module follow these steps:

(i) Locate the httpd.conf file within your Apache installation and open it for editing
(ii) Find the line as shown below and uncomment it by removing the preceding #

```
LoadModule rewrite_module modules/mod_rewrite.so
```

(iii) Find the line as shown below and uncomment it by removing the preceding # (it may already be uncommented)

```
ClearModuleList
AddModule mod_rewrite.c
```

2 Redirection Rules

The component can be programmed to carry out URL rewriting and redirection using the Perl Compatible Regular Expression (PCRE) syntax. The rewriting and redirection instructions are written in the `.htaccess` file. This file can be found at either site or server root level.

Using this syntax, it is possible to perform not only redirection but also URL rewriting. Also, you can carry out redirection not only to the web archive but to other locations as well. You should refer to the 'Readme' file for more details and to web resources such as www.pcre.org for help with the PCRE syntax.

The following examples give some basic examples of redirecting URLs to the Web Archive (webarchive.nationalarchives.gov.uk).

Note that you should substitute the web server name for `<mydomain>` in these examples.

2.1 Redirecting all un-resolved URLs

The following command redirects any unresolved URL to the archive site using a hard coded prefix for archive address.

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ http://webarchive.nationalarchives.gov.uk/ +/http://<mydomain>/$1
[R=301]
```

Request	http://www.nationalarchives.gov.uk/about/default.htm
Outcome	If the file is not found, redirected to: http://webarchive.nationalarchives.gov.uk/+/http://nationalarchives.gov.uk/about/default.htm

2.2 Redirecting all un-resolved URLs within a specific directory

The following command redirects unresolved requests relating to just a specific directory of a website – in this example, the 'about' directory under the root of The National Archives site.

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^about/(.*)$ http://webarchive.nationalarchives.gov.uk/
+/http://<mydomain>/about/$1 [R=301]
```

Examples:

Request	http://www.nationalarchives.gov.uk
Redirection	No redirection, because the URL does not meet the pattern specified in the Rewrite Rule. The home page of the site would be served to the user.

Request	http://www.nationalarchives.gov.uk/about/whowhathow.htm
Outcome	No redirection, because although the URL matches the pattern in the rewrite rule, it does exist on the web server.

Request	http://www.nationalarchives.gov.uk/about/nosuchpage.htm
Outcome	Redirected to: http://webarchive.nationalarchives.gov.uk/+/http://nationarchives/gov.uk/about/nosuchpage.htm

2.3 Redirecting specific URLs

This example redirects a single individual URL. The redirection occurs even if the URL exists on the server, because it is not preceded by the two `RewriteCond` lines in the earlier examples.

```
RewriteRule ^/tsoi.htm$   http:// www.tso.co.uk/tso.htm [R=301]
```

Example

Request	http://www.nationalarchives.gov.uk/tso.htm
Redirection	http://www.tso.co.uk/tso.htm

2.4 Capturing redirects back from the Web Archive

Where a page has been redirected to the web archive, the latest version of the archived page will be served back to the user if it is found in the archive.

If the page is not found, the web archive will issue a further redirection back to the user, as in the following example:

Request	http://www.nationalarchives.gov.uk/about/nosuchpage.htm
Redirection to web archive	http://webarchive.nationalarchives.gov.uk/+/http://nationarchives/gov.uk/about/nosuchpage.htm
Redirection from web archive	http://www.nationalarchives.gov.uk/ukgwacnf.html

You will need to handle these cases, in order to inform the user that the requested page was not found on either departmental web server or the web archive.

There are three options. The first is to have a page called ukgwacnf.html, which displays an appropriate message to the user. The second is to use URL rewriting to map these requests to another page which displays this information. The first rule in the sample ini file at Annex 3 demonstrates this approach. The third approach is to issue a redirect request from this resource to another page.

With the first and second approaches, the user will see http://<mydomain>/ukgwacnf.html in the browser address bar.

With the third approach, the browser address bar will show the location of the redirected page.

Section C: Ionics ISAPI Rewriter on Internet Information Server (IIS)

1 Installation of the Component

Ionic's Isapi Rewrite Filter (IIRF) is an open source component that can used for URL rewriting and redirection on Microsoft IIS installations. You can download it from http://www.codeplex.com/iirf, where you will also find additional documentation, a discussion forum and version information.

The `readme` file included in the download contains full instructions. There is also documentation available at http://cheeso.members.winisp.net/Iirf12Help/frames.htm, which, although it refers primarily to version 1.2.16, is also largely applicable to the earlier versions (discussed below). The latest version - 2.0 - has an MSI based installation package, with further details and documentation at http://cheeso.members.winisp.net/Iirf20Help/frames.htm.

The following is a summary of the installation process for versions 1.2.14 and 1.2.15.

1. The component can be installed on web sites hosted in IIS 5.0, 5.1 and 6.0 (Windows 2000, XP and 2003) by adding it to the ISAPI Filters tab in the Web Site Properties in Internet Information Services Manager
2. Place the Ionics DLL, IsapiRewrite4.dll, in an appropriate folder on the Web Server. For example this might be either

   `C:\wwwroot`

   or

   `C:\windows\system32\inetsrv`

   It is important not to place the DLL within the web document tree itself

3. If you are running multiple websites under a single IIS installation and want to use the facility on more than one site (and are using a version earlier than 2.0), there are two options. One is a single installation for all sites, with a common set of rewrite rules. The other is to install separate copies of the dll for each web server in different directories. With the introduction of version 2.0, it is possible to run multiple installations on a server with a separate ini file for each website.

4. An ini file `called IsapiRewrite4.ini` must be installed in the same directory as the dll (or in each directory with a copy of the dll, if more than one copy is installed for multiple websites).

   If you do not want to set up the rewrite rules at this stage, you can use the 'skeleton' ini at Annex 2.

   Alternatively, you can base your ini file on the example at Annex 3, or configure from one of the samples supplied with the product. If you want to enable logging, you should ensure that the specified RewriteLog location exists. If you don't want to use logging, you can use the # sign to comment out the RewriteLog and RewriteLogLevel lines.

5. Open the IIS management console and select the appropriate level for installing the component. For example, if a single installation will be used to control all websites hosted on this server select the 'Web Sites' node under the machine name in the tree.

   Alternatively, if the installation is to apply only to a specific site hosted on this server, select that site node from within the tree.

   Open the properties of the appropriate node (right click – properties)
   - select the ISAPI Filters tab and click 'Add'
   - enter a name for this instance of the filter, for example Ionics URL Rewriter, and then browse to the DLL as copied in step 1 above

6. Once configured, stop the IISADMIN server. There are two ways of doing this. Use the 'Services' applet that can be found within Administrative tools on the server or enter the following command at a command prompt:

```
net stop iisadmin /y
```

7. Finally, start the WWW Publishing Service. Once again this can be done using the 'Services' applet or by entering the following command at a command prompt:

```
net start w3svc
```

2 Redirection Rules

The component can be programmed to carry out URL rewriting and redirection using the Perl Compatible Regular Expression (PCRE) syntax. The rewriting and redirection instructions are written in the ini file. Using this syntax, it is possible to perform not only redirection but also URL rewriting. Also, you can carry out redirection not only to the web archive but to other locations as well. You should refer to the `readme` file for more details and to web resources such as www.pcre.org for help with the PCRE syntax.

The following examples cover the basics of redirecting URLs to the Web Archive. More complex examples can be found in the `readme` file, at Annex 3, and via the discussion at http:www.codeplex.com/iirf/ Thread/List.aspx. Note that you should substitute the web server name for `<mydomain>` in these examples.

Redirecting all un-resolved URLs

The following command redirects any unresolved URL to the archive site using a hard coded prefix for archive address.

Version 1.2.15

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RedirectRule ^(.*)$ http://webarchive.nationalarchives.gov.uk/ +/http://<mydomain>/$1
[R=301]
```

Version 1.2.14

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ http://webarchive.nationalarchives.gov.uk/ +/http://<mydomain>/$1
[R=301]
```

Request	http://www.nationalarchives.gov.uk/about/default.htm
Outcome	If the file is not found, redirected to: http://webarchive.nationalarchives.gov.uk/+/http://nationalarchives.gov.uk/about/default.htm

Redirecting all un-resolved URLs within a specific directory

The following command redirects unresolved requests relating to just a specific directory of a website – in this example, the 'about' directory under the site root.

Version 1.2.15

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RedirectRule ^about/(.*)$ http://webarchive.nationalarchives.gov.uk/
+/http://<mydomain>/about/$1 [R=301]
```

Version 1.2.14

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^about/(.*)$ http://webarchive.nationalarchives.gov.uk/
+/http://<mydomain>/about/$1 [R=301]
```

Examples:

Request	http://www.nationalarchives.gov.uk
Redirection	No redirection, because the URL does not meet the pattern specified in the Rewrite Rule. The home page of the site would be served to the user

Request	http://www.nationalarchives.gov.uk/about/whowhathow.htm
Outcome	No redirection, because although the URL matches the pattern in the rewrite rule, it does exist on the web server

Request	http://www.nationalarchives.gov.uk/about/nosuchpage.htm
Outcome	Redirected to: http://webarchive.nationalarchives.gov.uk/+/http://nationarchives/gov.uk/about/nosuchpage.htm

Redirecting specific URLs

This example redirects a single individual URL. The redirection occurs even if the URL exists on the server, because it is not preceded by the two `RewriteCond` lines in the earlier examples.

Version 1.2.15

```
RedirectRule ^/tsoi.htm$   http:// www.tso.co.uk/tso.htm [R=301]
```

Version 1.2.14

```
RewriteRule ^/tsoi.htm$   http:// www.tso.co.uk/tso.htm [R=301]
```

Example

Request	http://www.nationalarchives.gov.uk/tso.htm
Redirection	http://www.tso.co.uk/tso.htm

2 Capturing redirects back from the Web Archive

Where a page has been redirected to the web archive, the latest version of the archived page will be served back to the user if it is found in the archive.

If the page is not found, the web archive will issue a further redirection back to the user, as per the following example

Request	http://www.nationalarchives.gov.uk/about/nosuchpage.htm

Redirection to web archive	http://webarchive.nationalarchives.gov.uk/+/http://nationarchives/gov.uk/about/nosuchpage.htm
Redirection from web archive	http://www.nationalarchives.gov.uk/ukgwacnf.html

You will need to handle these cases, in order to inform the user that the requested page was not found on either departmental web server or the web archive.

There are three options. The first is to have a page called ukgwacnf.html, which displays an appropriate message to the user. The second is to use URL rewriting to map these requests to another page which displays this information. The first rule in the sample ini file at Annex 3 demonstrates this approach. The third approach is to issue a redirect request from this resource to another page.

With the first and second approaches, the user will see http://<mydomain>/ukgwacnf.html in the browser address bar. With the third approach, the browser address bar will show the location of the redirected page. Such a redirection can be implemented either via the IIS manager, or as a rule in the Ionics component.

In addition to the component and ini file, various other tools included with the component may prove helpful. The `TestDriver` programme allows you to test the rewrite rules against some sample ini files. The `IirfVersion` utility allows you to query the version of the component installed.

Finally, the `TestParse` facility allows you to check that the ini file is valid. More information on all of these is contained in the `readme` file.

3 Testing issues and Versions

Version 1.2.14 of the Ionics component has been extensively tested by The National Archives. The following issues were noted – these may also be relevant to the current release 1.2.15 but have not been tested on this version to the same extent:

- In some early testing, the redirection did not work with default documents (so that a request for a resource to the directory would redirect instead of going to the default document). However, this has not re-occurred in later stages of testing

- It is possible to use rewrite conditions (`RewriteCond`) statements to perform more complex processing than simply checking for the existence of files and directories. However, some issues with very long URLs were encountered when trying to match against the QUERY_STRING server variable. Therefore, it is recommended that rewrite conditions are not used for this purpose

- Problems can occur if the log file specified in the ini file contains spaces

- The redirection may not work for very long URLs. Current data suggests that the component, running on Windows 2003, can deal with URLs with query strings up to around 2000 characters

- We have had some issues regarding filter priority - in particular, when Ionics is installed at a website level and that website has ASP.Net applications underneath it. In this scenario calls made to aspx files may bypass Ionics because the ASP.NET framework operates at a web server level. A solution is to install the Ionics filter at the web server level and give it highest priority

In version 1.2.15, we noted:

- The documentation states that the [R] flag is only required when you want to use a redirection code other than 302. However, it was found that omitting the [R] code led to either a 404 or a 500 server error when attempting to redirect to another domain. On attempting to redirect to another location on the same domain, URL-rewrite type behaviour occurred (so the redirection target was served but the original URL was still displayed in the user's browser). Examination of the Ionics log file suggests that the component was attempting a standard rewrite rather than a redirection. Therefore, it is recommended that an [R] flag is specified for all Redirect Rules.

- Even when the [R] flag is specified, the response status description returned in the header was ‘OK’ under Windows XP, instead of the expected ‘Moved Permanently’ (301) and ‘See Other’ (303). On Windows 2003, the code alone was returned without a status description

The latest release as of this update is Version 2.0. It contains a number of enhancements, notably:

- separate ini files for each web site when installed for multiple sites on a web server
- a RewriteHeader directive to allow rewriting of request headers (this was originally introduced in Version 1.2.16)
- the ability to act as a reverse proxy for selected URLs

This version has not been tested or assessed by The National Archives. Further information is available at: http://cheeso.members.winisp.net/iirf20Help/frames.htm

Section D: The National Archives HTTP Handler for Internet Information Server 5.0 and 6.0 (IIS)

1 Introduction

An alternative to the Ionics Rewriter, and other similar tools, for IIS 5 and 6 environments is the custom handler page solution created by The National Archives' Systems Development Team. This installation consists of a customised aspx error page, which works as a handler for 404 errors raised by IIS. The redirection behaviour is specified via.NET web.config files. Version 2.0 (or later) of Microsoft's .NET framework must be installed for successful operation.

The fact that redirection behaviour can be triggered specifically on 404 errors means that this component is suited to installations where URI requests may not always be mapped to a physical file on the web server (in the approach using Ionics above, the component checks whether the file physically exists before redirecting). The installation can be configured to allow a default redirection at web server level, which can be overridden at directory level if required. However, there is not the same degree of versatility as with Ionics, for instance, it is not possible to specify complex pattern matches to the same extent.

It is also possible to run a mixed environment comprising this HTTP handler and the Ionics component. The handler could be used to specify a default redirection behaviour for the website, with the Ionics component being used to override this behaviour or set more complex processing on individual directories.

The installation procedure described below is based on The National Archives own experience, and is not intended as a definitive guide.

2 Installation of the Component

The first step is to select (or create) the target installation directory for the custom handler. This must be a specific application directory located virtually under the website root. The name `customerrors` is suggested for this application though you can choose a different name if you wish.

To set up the application directory, first decide upon and create the physical directory on the file system such as `D:\MyWebsite\customerrors\`. If this directory is not under the physical website root, you then need to set it up as a virtual directory.

1. right-click on the website node and select New-Virtual Directory
2. enter the alias name, for example `customerrors` and select the physical file location
3. in the next screen, select the physical path you decided on as above
4. in the 'Access Permissions' screen, you need to have 'Read' and 'Run Scripts' selected as a minimum – other options may be selected depending on your installation requirements and security policies

Once you have created the application directory, it will appear as an application under the website root node, for example as the `customerrors` application directory (fig 3).

Figure 3: Creation of application directory for redirection

Note that if the directory you've just created is not set up as an application (that is, it has a folder icon rather than the application icon in Figure 3 above), you need to make it an application. Do this by selecting the 'Virtual Directory' tab from the directory's properties, then clicking the 'Create' button next to the application name, and then specifying a name for the application in the 'Application Name' box (Figure 4 below). You also need to ensure that the .NET version for the directory is 2.0 or later (see Figure 1 above).

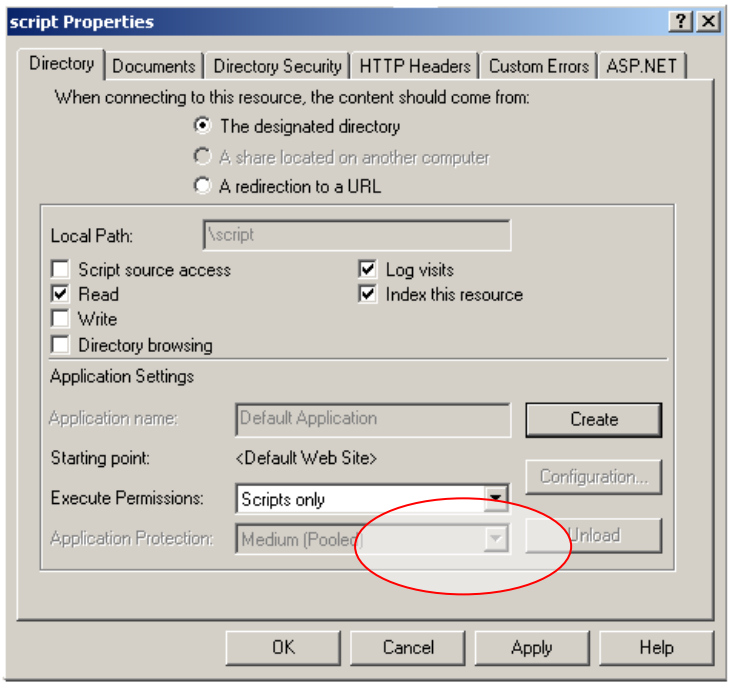

Figure 4: Creating an application in the IIS manager

Next, you need to ensure that the web application directory is running version 2.0 or later of the .NET framework. To do this, open the IIS Management Console, select the target server under the 'Web Servers' node, right-click, select properties, and then select the ASP.Net tab. You should see a screen similar to Figure 2 below:

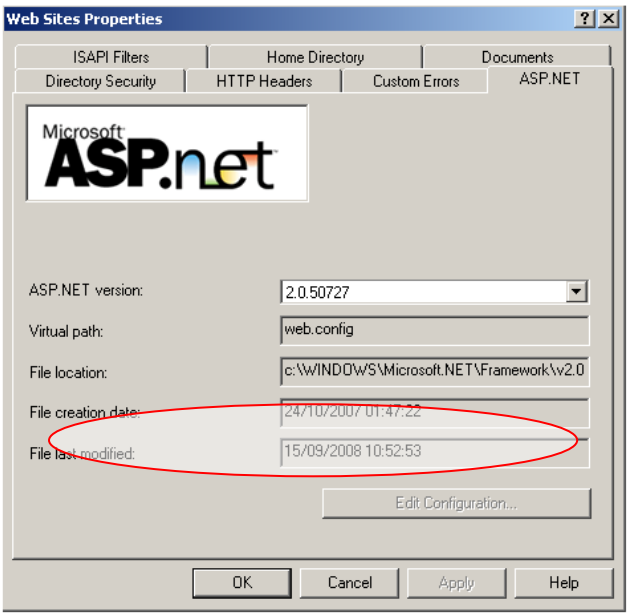

Figure 2: Setting the .NET version in IIS Manager

The ASP.Net version must be listed as 2.0 or above. If the version is lower than this, then you will be able to select a higher version if it is already installed on the computer, though you should be aware that doing so may lead to issues due to .NET version clashes and/or other problems elsewhere. Therefore, you should only change the version the change has been through your organisation's change management procedure first. If no later version is available, it can be downloaded from the Microsoft Website, along with installation and configuration instructions.

Note that the .NET version can be set at both web site and application level, and additional complications may occur where a web site or web server is running more than one version.

By default, a new web application will inherit the .NET version for the parent website, although it can be changed to a different version afterwards. For a web server running multiple websites, the .NET version can also be set at the Websites node level in addition to the website and web application level.

A further issue to address is the application pool in which the handler will run. All web applications in the IIS installation will be assigned to an application pool.

For simple installations, there may be only one application pool but more complex setups will have multiple pools to which individual web applications can be assigned.

Problems can occur when the `customerrors` application is assigned to handle errors from applications in other application pools, leading to the message: 'The specified request cannot be executed from current Application Pool'. To get around this, you either need to install multiple instances of the custom handler for each application pool, or add the IgnoreAppPoolForCustomErrors, registry key with a DWORD value of 1 to the w3svc service node in the system registry.

Full instructions on this are at: http://technet.microsoft.com/en-us/library/bb878092.aspx

Once the above steps are completed, you need to edit the web.config file (or create the file if not already there) for the website to ensure that aspx page requests are redirected to the custom handler. This is required under IIS 5.0 and 6.0 installations because such requests will be handled by the .NET runtime before being passed to the web server.

```
<customErrors mode="On">
  <error statusCode="404" redirect="/customerrors/404.aspx" />
</customErrors>
```

If there is no web.config file, this will need to be created from scratch. The following is the minimal file required to support redirection. It may also be necessary to amend the `<customErrors>` settings in this way for other application directories further down the virtual tree, if they have existing settings that override those required for redirection.

```
<configuration>
<system.web>
<customErrors mode="On">
  <error statusCode="404" redirect="/customerrors/404.aspx" />
</customErrors>
</system.web>
</configuration>
```

You will notice in the above example that a redirect parameter is specified in the `<error>` element. This points to the location of the handler page for 404 (Not Found) errors. In this illustration, the default path and name is assumed though you can change these if desired. You'll install this page in the next step.

Once the IIS configuration is complete, the 404.aspx page can simply be copied to the *customerrors* directory. Also, the associated dll (NationalArchives.Web.CustomHttpStatusErrors.dll) will need to be copied to the customerrors/bin sub-directory (this will need to be created if it doesn't exist already). This dll in turn references the following dlls in the Microsoft.Practices namespace, which must be copied to the bin sub-directory also:

```
Microsoft.Practices.ObjectBuilder.dll
Microsoft.Practices.EnterpriseLibrary.Common.dll
Microsoft.Practices.EnterpriseLibrary.Data.dll
```

`Microsoft.Practices.EnterpriseLibrary.Logging.Database.dll`

`Microsoft.Practices.EnterpriseLibrary.Logging.dll`

The next step is to configure the web.config file in the *customerrors* directory to set up the redirection rules. These are specified in the `<RedirectUris>` section, which is contained within the `<system.web>` section of the web.config file. The following is an example extract from a configuration file set up to specify a default redirection destination along with alternative destinations for specific directories.

```
<system.web>

……

<RedirectUris
defaultRedirectUri="http://dev.nationalarchives.tna.local/BridgingPage/BridgingPage.a
spx?url={0}" defaultResponseStatusCode="301" defaultReturnUri="testreturn.html">

    <items>

      <add requestUri="<servername>/test7/"
redirectUri="http://<servername>/test1/{0}" responseStatusCode="301" />

      <add requestUri="<servername>/test2/"
redirectUri="http://www.a2a.org.uk/?uri={0}" responseStatusCode="301" />

      <add requestUri="<servername>/test6/"
redirectUri="http://<servername>/test7/{0}" responseStatusCode="301" />

      <add requestUri="<servername>/place6/"
redirectUri="http://www.learningcurve.gov.uk?{0}" responseStatusCode="301"
substituteRequestUriString="False" />

      <add requestUri="<servername>/place7/"
redirectUri="http://<servername>/academic/{0}" responseStatusCode="301"
substituteRequestUriString="True" />

    </items>

  </RedirectUris>

<system.web>
```

To go through the details of this, the opening element is `<RedirectUris>`. It has the following attributes:

Attribute Name	Purpose
`defaultRedirectUri`	Required. Specifies the default redirection URI for the web server. Can optionally include the {0} placeholder which allows either the entire original URI or just the filename portion to be included in the redirected URI, depending on the `fileNameOnly` setting (see below). For the web archive, the correct entry is `http://webarchive.nationalarchives.gov.uk/+/{0}` However, at The National Archives an intermediate 'bridging page' is used rather than a direct forwarding to the web archive. The user can then manually navigate to the latest archive version from this page.
`defaultResponseStatusCode`	Optional. Specifies the HTTP code to be returned with redirection. If omitted, the default code 302 is returned. If redirection occurs to a 'bridging page' similar to that implemented by The National Archives (see below), a code of 303 ('See Other') is suggested
`defaultReturnUri`	Required. The default return redirection, or 'archive checked' URI, to which users will be redirected from the destination server specified in `defaultRedirectUri` if the requested page is not found there. The European Archive is currently configured to specify ukgwacnf.html for this purpose. This allows a special 'archive checked' URL to be configured in order to advise users that the requested URI was not found either on the original site or in the web archive. Note that the return URI is actually determined by the web archive so this setting does not actually force use of the specified page. However, it does ensure that the specified page is not processed by the handler as a standard 404 error, thus avoiding a

	circular redirection. Although you can make ukgwacnf.html your actual 'archive checked' page, you should serve a 404 error in this situation, and this is not easy to do if you serve this page directly. One way to achieve this is to have an asp/aspx page coded to return a 404 error, and redirect to this page from ukgwacnf.html. This redirect can be set up from within the IIS management console. The National Archives 'bridging page' (discussed below) pre-checks the web archive for the existence of the requested URL, and therefore the return URI specified is likely to be requested only in limited circumstances.

At this stage, the default configuration is complete. However, you may want to override the default behaviour for specific URIs or directories, in order to redirect certain requests to a destination other than the web archive. This is achieved by adding `<add>` elements to the `<items>` collection. The attributes in each `<add>` element function as follows:

Attribute Name	**Purpose**
`requestUri`	String to specify URL(s) for which this redirection behaviour will override the default behaviour. This can be either a complete URI or part of a URI, such as to specify the behaviour for all URIs in a specific directory.
`redirectUri`	Required. Specifies the redirection URI for all URLs which start with the string specified in `requestUri`. Can optionally include the {0} placeholder which allows either the entire original URI or just the filename portion to be included in the redirected URI, depending on the `fileNameOnly` setting (see below).
`responseStatusCode`	Required. Specifies the HTTP status code to be returned with redirection. Normally this will be code 301 (Moved Permanently) or code 302 (Moved Temporarily) or 303 (See Other).

substituteRequestUriString	Optional, with a default of True. Specifies whether the entire original URI (including the protocol) or just the filename (true setting) should be substituted for the {0} placeholder in redirectUri. Normally this attribute would be set to true (or omitted) for specific redirections to destinations other than the web archive, for example the following settings: requestUri= "www.myserver.gov.uk/oldsource" redirectUri="http://www.myserver.gov.uk/newsource/{0}" will facilitate redirection from: http://www.myserver.gov.uk/oldsource/myfile.html to: http://www.myserver.gov.uk/newsource/myfile.html

By default, the handler will redirect all file types matching the rules. This can lead to 'invisible' redirects where resources called within a page are not found and secondary redirection occurs in addition to the main one which is visible to the user. This can impact on web server performance, and can be avoided by excluding certain file types from the handler. Such redirections are also usually superfluous, since a web page arrived at via redirection will then fetch its own images and related files from URLs relative to its redirected location. The following entry in the web.config file for the customerrors application is used to exclude css and inc extensions from the redirection handler at The National Archives.

```
<applicationSettings>
  <NationalArchives.Web.CustomHttpStatusErrors.Properties.Settings>
    <setting name="LoggingTitle" serializeAs="String">
      <value>Custom Errors Logger</value>
    </setting>
    <setting name="ExcludeFileTypeExtensions" serializeAs="Xml">
      <value>
        <ArrayOfString xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
            xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <string>css</string>
            <string>inc</string>
          </ArrayOfString>
        </value>
      </setting>
    </NationalArchives.Web.CustomHttpStatusErrors.Properties.Settings>
  </applicationSettings>
```

The final stage in configuring the custom handler is to specify the customerrors/404.aspx page as the ‘handler’ for 404 errors. This would normally be done at the website level. Once again, this is available via the website properties – this time, on the ‘Custom Errors’ tab (Figure 5 below). You need to specify the handler for 404 errors, setting the Message Type to URL (Figures 5 and 6 below).

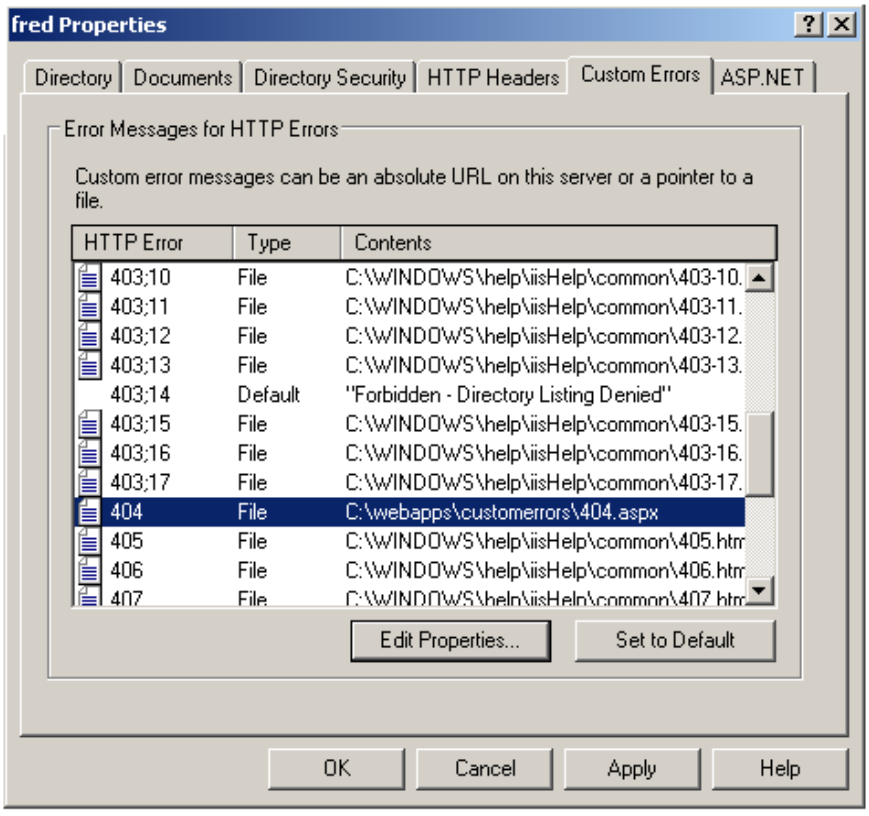

Figure 5: Customer error handlers dialogue box

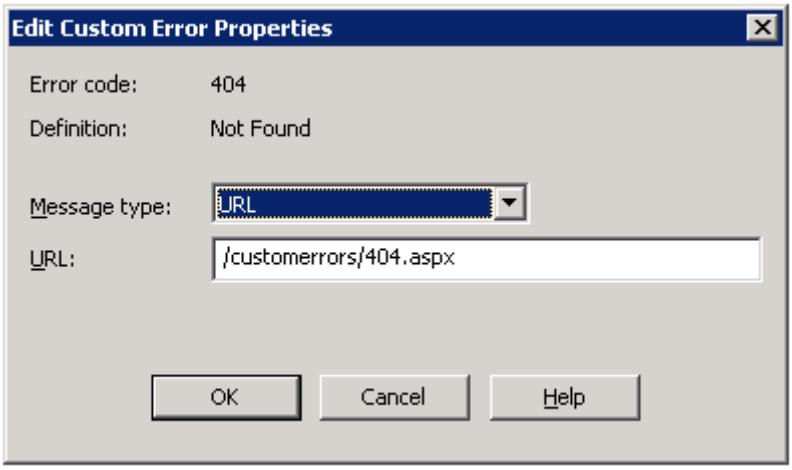

Figure 6: Specifying the customer error handler for 404 errors

3 Bridging Page

As described above, a default redirection is specified for resources not found, and this allows such requests to be forwarded directly to the web archive. However, web managers may prefer that users are redirected to an intermediate or 'bridging page', instead of being sent direct to the web archive.

Whilst most pages in the web archive contain a banner at the top to indicate that the page is archived content, as well as a modified title tag, this may not always be sufficient to alert the user that they have jumped from a live site to the web archive.

As well as the main redirect to the web archive, bridging pages can also be used for secondary redirects if desired (though the current The National Archives implementation is for the web archive only). A bridging page also allows the user to choose whether or not to follow the link to the web archive or other source. On the other hand, a using a bridging page means that any search engine ranking applicable to the original page is not transferred to the archived version.

The National Archives bridging page is implemented as a further .NET web application, and consists of a single compiled aspx page, web.config file and accompanying compiled bin directory in a dedicated web application directory.

In The National Archives implementation, the page checks whether the originally requested URL exists in the web archive, by issuing a HTTP HEAD request. If the page is found to exist, the page is displayed to the user in 'bridging page' mode, and will include a link to the archived

version of the URL that the user can then follow. If the URL is not found in the archive, the page will display in ‘not found’ mode rather than ‘bridging page’ mode. There will be no link to the web archive and the page will be similar to a standard 404 page. The return code in this case will be 404 – Not Found, whereas the return code in ‘bridging page’ mode is 200.

If an error occurs in the pre-display HEAD request to the web archive, for example due to incorrect proxy settings or no response from the archive, it is assumed the page does exist in the archive and the display is in ‘bridging page’ mode. However, error reporting can be configured using the .NET health checking capabilities to inform technical staff about the problem.

The web.config file has the following entries under `<appsettings>`.

```
      <appSettings>
    <add key="sourceDomains"
value="test.myserver.gov.uk,www.myserver.gov.uk" />
    <add key="defaultRedirectLinkTarget" value="http://test.myserver.gov.uk" />
    <add key="webArchiveLocation"
value="http://webarchive.nationalarchives.gov.uk/+/"/>
    <add key="Archive_Checked_Url" value="ukgwacnf.html"/>
    <add key="Request_Timeout" value="30000"/>
    <add key="Max_Url_Length" value="1024"/>
    <add key="Found_Return_Code" value="200"/>
    <add key="NOT_Found_Return_Code" value="404"/>
    <add key="Enable_Archive_Check" value="true"/>
    <add key="Proxy_UserName" value="[username]"/>
    <add key="Proxy_Password" value="[password]"/>
  </appSettings>
```

Attribute Name	Purpose
`sourceDomains`	Comma separated list of permitted referring domains – see below.
`Default_Redirect_Link_Target`	Legacy setting – may be removed in later release as no longer required. Advise setting to your website domain name.
`webArchiveLocation`	Location of the web archive. This will be used in when constructing the link to the web archive when the page is displayed in 'bridging page' mode.
`Archive_Checked_Url`	The return URL used by the European Archive to redirect when a request is not resolved in the archive. Should be set to ukgwacnf.html
`Request_Timeout`	Timeout value for web archive HEAD check. If the timeout is exceeded, the page displays in Bridging Page mode as if the check had returned a positive result. Value is in milliseconds, for example 30000 means 30 seconds.
`Max_Url_Length`	The maximum URL length that will be fully processed by the handler. If this is exceeded, the page display in 'Not Found' mode without further processing. Enter 0 for no maximum length restriction.
`Found_Return_Code`	The HTTP code to return when the page displays in 'Bridging Page' mode, for example 200
`NOT_Found_Return_Code`	The HTTP code to return when the page displays in 'Not Found' mode, for example 404
`Enable_Archive_Check`	When set to `false`, the web archive HEAD pre-check is disabled and the page goes straight to 'Bridging Page' mode as if the HEAD request returned positive. This setting is recommended as a temporary measure in certain situations, for example network problems, or load testing where this may feed through to an increased load on the web archive infrastructure.

Proxy_UserName	User name for the proxy server through which the page needs to access the internet.
Proxy_Password	Password for the proxy server through which the page needs to access the internet.
Proxy_UserDomain	Domain for the proxy server through which the page needs to access the internet.

The location of the proxy server is specified in the <system.net> section, as follows:

```
<system.net>
  <defaultProxy>
    <proxy proxyaddress="http://<proxy-server>:<proxy-port>" bypassonlocal="True"/>
    <bypasslist>
      <add address="[a-z]+\.myserver\.my.domain"/>
    </bypasslist>
  </defaultProxy>
</system.net>
```

The value attribute of the sourceDomains key contains a comma separated list of allowed domains. When a domain is included in this list, it can 'set' the web archive page link from the bridging page (see Figure 7 below). For example assume that with the above configuration, the following URL is generated by the custom error handler's default redirection setting:

http://www.myserver.gov.uk/BridgingPage/BridgingPage.aspx?url=http://www.myserver.gov.ukNoSuchPage.aspx

This will call the bridging page, and the 'Check for this Page in the Web Archive' link will point to:

http://webarchive.nationalarchives.gov.uk/+/http://www.myserver.gov.ukNoSuchPage.aspx

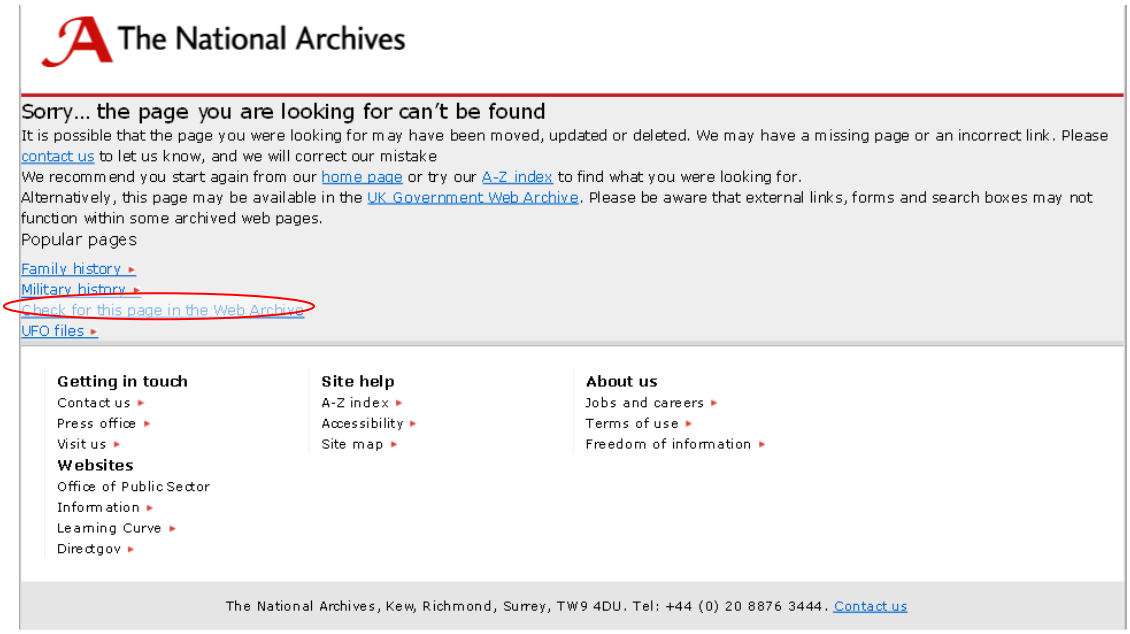

Figure 7: Bridging Page, with link to web archive page highlighted

If the domain is not included as above in the list of permitted domains, the bridging page will display in 'Not Found' mode. This is to prevent the application being used by malicious persons to redirect the user to an undesirable site by making use of the return redirection to `<domain-name>/ukgwacnf.html` described in (2) above.

For example without this check, a user could set up the URL http://www.mydodgysite.co.uk/ukgwacnf.html

and send users to it via the following, thus faking a legitimate referral from the department:

http://test.mysite.gov.uk/PageNotFound/PageNotFound.aspx?url=http://www.mydodgysite.co.uk/intro.htm

The handler is designed to continue operating where certain error conditions occur. For example if network issues prevent the HEAD pre-check from working, or the web archive returns an unexpected error code, for instance, 404. In such cases, the display defaults to 'Bridging Page' mode as if the HEAD pre-check had returned positive. However, a custom error is raised by the application for information. The processing of these errors can be controlled via .NET health check settings in the web.config file. As an example users may want information to be emailed to a specified mailbox and/or logged in the Windows event log. The settings can be used to control the number and frequency of notifications sent, which is useful in situations that can generate a

large flow of similar error messages. See the .NET documentation for further information on health checking.

1. Known Issues:

(i) Each time a redirection rewrite take place, the rewritten URL is evaluated once again against the entire set of rules, irrespective of the order in which they are listed. In the following example

http://www.mydepartment.gov.uk/test5/default.htm

http://www.mydepartment.gov.uk/test6/default.htm

http://www.mydepartment.gov.uk/test7/default.htm

will all redirect to:

http://www.mydepartment.gov.uk/test8/default.htm

```
<RedirectUris defaultRedirectUri="http://webarchive.nationalarchives.gov.uk/+/{0}"
defaultResponseStatusCode="301" defaultReturnUri="ukgwacnf.html">

    <items>

      <add requestUri="www.mydepartment.gov.uk/test7/"
redirectUri="http://www.mydepartment.gov.uk/test8/{0}" responseStatusCode="301"  />      <add
requestUri="www.mydepartment.gov.uk/test5/" redirectUri="http://www.mydepartment.gov.uk/test6/{0}"
responseStatusCode="301"  />

      <add requestUri="www.mydepartment.gov.uk/test6/" redirectUri="http://
www.mydepartment.gov.uk/test7/{0}" responseStatusCode="301"   />



    </items>

  </RedirectUris>
```

(ii) You should avoid situations where a URL will meet more than one requestUri in a single pass. This can lead to unexpected results, including an infinite redirection loop.

(iii) Issues with application pools and the .NET version, as discussed above.

Annex 1: Sample .htaccess file for Apache mod_rewrite

```
# N.B. # indicates the start of a comment line - this may not always # appear due to word wrapping settings in your viewing software

# Ensure rewriting is turned on

RewriteEngine On

# 1.  CAPTURE REDIRECTS BACK FROM THE WEB ARCHIVE

#

# If a request has previously been redirected to the Web Archive, and the page was not found there,

# the web archive will issue a redirect back to a page called ukgwacnf.html.  The following rule rewrites

# this to a custom "Archive Checked, document not found" page, and then stops all further processing.

# Note there is no need for ukgwacnf.html to actually exist - this rule will operate regardless.  Also,

# the browser URL will show [domainname]/ukgwacnf.html but the content will be from 404webarchive.asp

# The [L] flag here stops any further processing of the request by further rules below, if it is caught

# by this rule.


RewriteRule ^/ukgwacnf.html /404webarchive.asp [L]


# 2.  CAPTURE ALL URLs OVER A SPECIFIED LENGTH

#

# This rule redirects requests to a handler page all requests where the non-domain portion of the URL is

# 100 or more characters.  This can be useful in trapping malicious server traffic (in reality, the lower

# threshold length may be somewhat above 100).
```

```
RewriteRule ^/(.{100,})  /blank.html [R=301]


# 3.  RE-WRITE URLS FOR A SPECIFIC DIRECTORY IF FILE NOT FOUND

#

# In this rule, for any request where the top level directory is "newdir", and the
document does not exist,

# the URL is rewritten substituting "olddir". i.e.  the URL in the browser will still
appear with "newdir"

# but the resource will come from the olddir directory instead, as if the user had
entered the "olddir"

# version of the URL.  Processing then continues with further rules.


RewriteCond %{REQUEST_FILENAME} !-f

RewriteCond %{REQUEST_FILENAME} !-d

RewriteRule ^/newdir/(.*)$    /olddir/$1


# 4. .HTML to .HTM

#

# If the user has entered a file with a .htm extension and the file is not found,
rewrite the file extension

# to .html, then continue processing with further rules.


RewriteCond %{REQUEST_FILENAME} !-f

RewriteRule ^/(.*).html$   /$1.htm


# 5. REDIRECT SPECIFIED DIRECTORIES TO WEB ARCHIVE

#

# Where the top level directory is documentsonline, search, test1, ionics or
academic,

# and the file does not exist, the request is redirected to the web archive.  E.g.
```

```
# http://www.nationalarchives.gov.uk/documentsonline/nosuchfile.asp is redirected to:

#
http://webarchive.nationalarchives.gov.uk/tna/+/http://www.nationalarchives.gov.uk/do
cumentsonline/nosuchfile.asp



RewriteCond %{REQUEST_FILENAME} !-f

RewriteCond %{REQUEST_FILENAME} !-d

RewriteRule ^/(documentsonline|search|test1|ionics|academic)/(.*)$
http://webarchive.nationalarchives.gov.uk/tna/+/http://www.nationalarchives.gov.uk/$1
/$2 [R=301]



# N.B. To redirect everything that reaches this point to the web archive, replace the
above rewrite rule with the following

# one (just place a # comment marker before the rule above and remove the # from the
rule below).



#RewriteRule ^(.*)$
http://webarchive.nationalarchives.gov.uk/tna/+/http://www.nationalarchives.gov.uk/$1
[R=301]
```

Annex 2: ‘Skeleton’ ini file for Ionics Rewriter

```
# N.B. # indicates the start of a comment line - this may not always # appear due to word  wrapping settings in your viewing software

# IsapiRewrite4.ini

#

#

# comment lines begin with #

#

# These are the currently  supported directives for the ini file:

#

#  IterationLimit

#  MaxMatchCount

#  RewriteRule

#  RewriteLog

#  RewriteLogLevel

#  RewriteCond

#

#

# Check the examples below for how to use each one.

# See the Readme.txt for full explanation of the rules here.

#


#N.B. Logging will only work if this directory exists and the account running IIS
#has Read/Write permissions on the directory.  Also, there appear to be an

#issue with spaces in the log file, so it’s best to avoid these.

RewriteLog  c:\ionics\log\iirfLog.out

RewriteLogLevel 3
```

```
# MaxMatchCount

#

# Specifies the maximum number of sub-expression matches to

# capture for a single pattern. This specifies the size of the

# array in the C module.  If you have a pattern with more than

# the default number of matches, set this number.

#

# The default is 10.


MaxMatchCount 10


#Rewrite rules follow  here….
```

Annex 3: Sample ini file for Ionics Rewriter

 # N.B. # indicates the start of a comment line - this may not always # appear due to word wrapping settings in your viewing software

```
# IsapiRewrite4.ini

#

# sample ini file for the ISAPI rewriter.

#

# comment lines begin with #

#

# These are the currently  supported directives for the ini file:

#

#  IterationLimit

#  MaxMatchCount

#  RewriteRule

#  RewriteLog

#  RewriteLogLevel

#  RewriteCond

#

#

# Check the examples below for how to use each one.

# See the Readme.txt for full explanation of the rules here.

#

#



#Looks like you can't have spaces in the log file location

RewriteLog  C:\Ionics\log\iirfLog.out

#
```

```
#Specify the rewrite log level (1-5).  This should normally be no

#higher than 3 for production systems.

RewriteLogLevel 3


# MaxMatchCount

#

# Specifies the maximum number of sub-expression matches to

# capture for a single pattern. This specifies the size of the

# array in the C module.  If you have a pattern with more than

# the default number of matches, set this number.

#

# The default is 10.


MaxMatchCount 10


# This example ini file is based on version 1.2.15 of the Ionics

# rewriter.  If using version 1.2.14, use RewriteRule for both      # rewrites and
redirections (with appropriate the R flag) rather

# than using RedirectRule for redirections.


# ***  1.   CAPTURE REDIRECTS BACK FROM THE WEB ARCHIVE *****

# ********************************************************

# If a request has previously been redirected to the Web Archive,   # and the page
was not found there, the web archive will issue a

# 301 redirect back to a page called ukgwacnf.html.  The following  # rule rewrites
this to a custom "Archive Checked, document not

# found" page, and then stops all further processing.

# Note there is no need for ukgwacnf.html to actually exist - this  # rule will
operate regardless.  Also, since this is a rewrite

# rather than a redirect,the browser URL will show
```

```
# [domain-name]/ukgwacnf.html but the content will be from

# 404webarchive.asp

# The [L] flag here stops any further processing of the request by

# further rules below, if it is caught by this rule.


RewriteRule ^/ukgwacnf.html /404webarchive.asp [L]




# *** 2.    CAPTURE ALL URLs OVER A SPECIFIED LENGTH

# ********************************************************

# This rule redirects requests to a handler page all requests where # the URL is 100
or more characters long.  This can be useful in

# trapping malicious server traffic (in reality, the lower

# threshold length may be above 100.  Also, you may want to consider

# only the non-domain name part of the URL, or the query string, in # which case a
more complex expression would be needed).


RedirectRule ^/(.{100,}) /blank.htm [R=301]

# *** 3. .HTML to .HTM

# ********************************************************

# If the user has entered a file with a .html extension and the file # is not found,
rewrite the file extension to .htm, then continue

# processing with further rules.


RewriteCond %{REQUEST_FILENAME} !-f

RewriteRule ^/(.*).html$   /$1.htm

-


# *** 4.    RE-WRITE URLS FOR A SPECIFIC DIRECTORY IF FILE NOT FOUND #
*****************************************************
```

```
# In this rule, for any request where the top level directory is
# "about", and the document does not exist, the URL is rewritten
# substituting "academic". i.e.  the URL in the browser will still
# appear with "about" but the resource will come from the
# academic directory instead, as if the user had entered "academic"

RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^/about/(.*)$     /academic/$1

# *** 5. REDIRECT SPECIFIED DIRECTORIES TO WEB ARCHIVE
# ******************************************************
# Where the top level directory is "documentsonline", "search", or
# "dambusters" and the file does not exist, the request is
# redirected to the web archive.  E.g.
# http://www.nationalarchives.gov.uk/documentsonline/about.asp
# would (if it did not exist) be redirected to:
# http://webarchive.nationalarchives.gov.uk/tna/+/
# http://www.nationalarchives.gov.uk/documentsonline/about.asp

RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RedirectRule ^/(documentsonline|search|dambusters)/(.*)$

http://webarchive.nationalarchives.gov.uk/tna/+/http://www.nationalarchives.gov.uk/$1
/$2 [R=301]

# N.B. To redirect everything that reaches this point to the web
# archive, replace the above rewrite rule with the following
```

```
# one (just place a # comment marker before the rule above and

# remove the # from the rule below).


#RedirectRule ^(.*)$  #http://webarchive.nationalarchives.gov.uk/tna/+/

#http://www.national#archives.gov.uk/$1 [R=301]
```