

***SFA Modernization Partner***  
**United States Department of Education**  
**Student Financial Assistance**



# **SFA Portal Strategy Integration Results**

***Task Order #48***

November 2, 2001

# TABLE OF CONTENTS

<b>1</b>	<b>EXECUTIVE SUMMARY .....</b>	<b>3</b>
1.1	INTRODUCTION .....	3
1.2	APPROACH.....	3
1.3	PHASE I.....	3
1.4	PHASE I RESULTS .....	4
1.5	PHASE II.....	5
1.6	PHASE II RESULTS .....	5
<b>2</b>	<b>PORTAL COMPONENTS DEPLOYMENT .....</b>	<b>5</b>
2.1	OVERVIEW.....	5
2.2	INVENTORY.....	6
2.3	DIRECTORY STRUCTURES.....	7
2.4	DEPLOYMENT .....	8
<b>3</b>	<b>PORTAL COMPONENTS INTEGRATION .....</b>	<b>8</b>
3.1	PREREQUISITES .....	8
3.2	INTEGRATION.....	8
3.3	TEST RESULTS .....	9
<b>4</b>	<b>TECHNICAL RECOMMENDATIONS .....</b>	<b>9</b>
<b>5</b>	<b>ARCHITECTURAL RECOMMENDATIONS.....</b>	<b>10</b>
5.1	APACHE STRUTS – PROS.....	10
5.2	APACHE STRUTS – CONS .....	10
5.3	WEBSHERE PORTAL SERVER CONSIDERATIONS .....	11
<b>6</b>	<b>APPENDIX A – FILE LIST .....</b>	<b>11</b>
6.1	SQL DATABASE SCRIPTS .....	11
6.2	JAVA SOURCE.....	11
6.3	HTMLS.....	12
6.4	JSPS .....	13
6.5	GIFS.....	14
6.6	CONFIGURATION FILES.....	15
<b>7</b>	<b>APPENDIX B – DEPLOYMENT INSTRUCTIONS .....</b>	<b>15</b>
<b>8</b>	<b>APPENDIX C – TEST PLAN.....</b>	<b>15</b>

## 1 Executive Summary

### 1.1 Introduction

The U.S. Department of Education, Office of Student Financial Assistance (SFA) has endeavored to create a new Portal Strategy. Based on examination of their existing Schools Portal application, SFA outlined the following objectives for the new strategy:

- Remove the use of all Viador components.

- Migrate to the WebSphere development architecture.

- Restructure the Schools Portal application into a modular application design consistent with best practice principles.

- Develop seven (7) reusable portal components based on the following functions: Registration, Login, Personalization, Search, Calendar, Feedback, and Headlines.

- Incorporate Reusable Common Services (RCS) functionality as applicable.

Accomplishment of these steps would provide standard functionality for reuse in the development of future SFA portal applications, compatible with the current architectural enterprise.

### 1.2 Approach

An obvious approach to accomplishing the aforementioned objectives was to modify the existing Schools Portal application structure and code, already in production. Determining the feasibility of this approach required some analysis of the existing Schools Portal application architecture and code. Upon doing so, it became evident that the existing site did not possess the necessary foundation for developing reusable portal components. There was little in the existing structure or code of the Schools Portal that could be reused toward these objectives.

The technical architecture team investigated other approaches such as using a commercial off the shelf portal, customizing a portal using some commercial off the shelf portal components or creating a lightweight portal. After careful review, creating a lightweight portal by using the open source framework of Apache Struts was chosen as the best architectural approach.

The technical development team used what they could glean from the existing production site as a baseline for their development effort. After reviewing the seven functions selected from the Schools Portal application and the open source framework approach, the development team defined a two-phase development approach, as outlined below.

### 1.3 Phase I

In Phase I, the technical development team wanted to accomplish the following tasks:

- Remove the use of all Viador components.

- Migrate to the WebSphere development architecture.

- Restructure the Schools Portal application into a modular application design consistent with best practice principles.

Create a lightweight portal utilizing the open source framework of Apache Struts.

Some of the high-level steps to accomplish these tasks included designing each portal component, establishing the framework for the lightweight portal, setting up a WebSphere development environment with Apache Struts, and then coding the portal components. Please see the project plan for the specific detailed steps.

#### 1.4 Phase I Results

The technical development team successfully accomplished these tasks. As a result, Apache Struts was added to the SFA development environment along with the portal component code for Phase I. The benefits gained through the accomplishment of these tasks is evident with a technical code analysis.

The following is just one of several examples of the benefits received by the portal component code for Phase I:

The existing SFA Schools Portal application displays a static calendar on the following six pages:

Home page	created in calendar.jsp
Events page	created in calendar_events.jsp
Deadlines page	created in calendar_deadlines.jsp
Training page	created in calendar_training.jsp
NPRMs page	created in calendar_nprms.jsp
All Categories page	created in calendar_allitems.jsp

The appearance of each calendar is more similar than not, yet each calendar is maintained in a separate file.

A comparison of the source code to produce the home page (ie. calendar.jsp) with the source code to produce the Events page (ie. calendar\_events.jsp) yielded a match of approximately 160 of 460 lines. While a comparison of the source code to produce the Events page (ie. calendar\_events.jsp) with source code to produce the other category pages yielded the following results:

Versus Deadlines (calendar\_deadlines.jsp) => a match of approximately 340 of 355 lines

Versus Training (calendar\_training.jsp) => a match of approximately 330 of 355 lines

Versus NPRMs (calendar\_nprms.jsp) => a match of approximately 325 of 355 lines

Versus All Items (calendar\_allitems.jsp) => a match of approximately 300 of 500 lines

The high numbers of matching lines indicate that multiple files are not really necessary, especially for the category pages. The approximately 160 matching lines of code within each of the six files probably relate to the calendar's appearance. An appearance change for one calendar means making the same change to the duplicate lines of code in the other files. This

type of structure within the existing application requires more development and maintenance cost than necessary.

To illustrate with a specific example, within each of these 6 files are approximately 50 lines of HTML to produce a part of the calendar appearance. This means not only 300 (6 x 50) lines to maintain but also 6 places to modify if the background color of the category choice bar should be changed to a different color.

Visually, the calendar appearance did not change between the Phase I code and the production site but its structure did. Within the new design is a total of approximately 60 lines of HTML with the increase to handle the few differences. This means not only an 80% reduction in the number of lines to maintain but also only one place to modify if the background color of the category choice bar needs to be changed. This creates less developmental and maintenance cost for the future.

## **1.5 Phase II**

In Phase II, the technical development team wanted to accomplish the following tasks:

- Develop reusable portal components.

- Incorporate Reusable Common Services (RCS) functionality as applicable.

Some of the high-level steps to accomplish these tasks included installing the RCS components into the existing development environment, designing the integration of the RCS components into the lightweight portal, coding the integration of the RCS components into the portal components, and then testing the system. Please see the project plan for the specific detailed steps.

## **1.6 Phase II Results**

The technical development team successfully accomplished the aforementioned tasks. As a result, the procedures for deployment and integration of the Portal Components have been documented in the following sections. In addition, sections have been included outlining possible technical and architectural recommendations. Also, a demonstration will occur to showcase the work that has been accomplished during the first phase of the SFA Portal Strategy.

# **2 Portal Components Deployment**

## **2.1 Overview**

The aforementioned broad objectives of SFA's Portal Strategy were refined into more specific tasks to be completed in phases. The tasks for this first phase were outlined as follows:

- Remove the use of all Viador and JRUN components.

- Migrate onto IBM HTTP and WebSphere Application Servers in development environment.

- Restructure into a modular application design consistent with best practice principles.

- Develop the following reusable portal components:

Registration – will provide the ability to register to the Schools Portal web site. A user will provide requested input and then will receive screen notification as to his/her user ID.

Login – will provide the ability to perform login and authentication of a registered user ID.

Personalization – will provide the ability for a registered, logged on user to personalize a home page that will appear each time the registered user logs on. The first time a registered user logs on, his/her home page will be presented identical to the Schools Portal main home page. The user will then have the ability to personalize the following:

Change password – will provide the ability for the user to change his/her password. Note: This feature does not include any password management capabilities.

Alter SFA links display – will provide the ability for the user to choose which SFA links to display on his/her home page.

Add/modify bookmarks – will provide the ability for the user to add and modify personal bookmarks (URLs of other web sites) that will be displayed on his/her home page.

Search – will provide an interface to the Autonomy search engine. This interface is currently implemented using CGI. The CGI implementation will be removed and replaced with an API.

Calendar – will provide the ability to display a static calendar on the Schools Portal web site or on a registered, logged on user's home page. The calendar data will be static and will not contain user-specific calendar information.

Feedback – will provide the ability to display a static page defining the process for reporting feedback via telephone or via email. The email functionality will be provided by the client browser configuration, and thus no server side email capabilities will be required.

Headlines – will provide the ability to display static headline data on the Schools Portal web site or on a registered, logged on user's home page.

Incorporate the functionality of the Reusable Common Services as applicable. Specifically, Exception Handling, Logging, Persistence and Search.

## 2.2 Inventory

The inventory necessary for the Portal Components development included the use of various vendor software components as well as some application-specific components.

The following vendor software components were used to develop the Portal Components:

Apache Struts 1.0

Autonomy Search Engine

Database - IBM DB2 7.2.3 or Oracle 8I DBMS 8.1.5  
IBM HTTP Server 1.3.6.3  
IBM Visual Age for Java 3.5.3  
IBM WebSphere Application Server 3.5.3  
Java Development Kit 1.2  
JDBC drivers (as shipped with Database product)  
SFA Reusable Common Services 1.0

The application-specific components developed by IBM Global Services for the Portal Components and Schools Portal application consists of approximately 11,885 source lines of code in 165 source files.

For a specific listing of these files, see Appendix A. These files can be categorized as follows:

- 59 GIF images
- 45 Java Source or Class files
- 28 Java Server Pages (JSPs)
- 23 HTML pages
- 19 Configuration Files
- 9 Struts Tag Library Definition (TLD) files
- 3 eXtensible Markup Language (XML) files
- 3 Struts Database Tag Definition (DTD) files
- 2 Cascading Style Sheet (CSS) files
- 2 Properties files
- 4 Oracle SQL database scripts

### 2.3 Directory Structures

The directories utilized for deployment of the Portal Components and/or the Schools Portal application are as follows:

DIRECTORY	CONTENTS
/www/dev/eip/sfa	SFA Portal Component directory
../sfa/build/scripts	SQL database script files
../sfa/servlets/sfa/portlet	Java Source and Class files
../sfa/web	HTML and JSP files
../sfa/web/images	GIF files
../sfa/web/WEB-INF	Configuration files

## 2.4 Deployment

This section documents the checklist to deploy the Portal Components and/or Schools Portal application into the development environment.

Install the following software products:

- Database (type is your choice)
- IBM HTTP and WebSphere Application Servers
- Apache Struts
- Autonomy Search Engine
- SFA Reusable Common Services
- SFA Portal Components Framework

Configure the aforementioned products.

For specific installation instructions, see the SFA Portal Installation Specification document in Appendix B.

## 3 Portal Components Integration

### 3.1 Prerequisites

The following hardware and software prerequisites are required prior to integration and/or usage of the Portal Components:

- Apache Struts 1.0
- Autonomy Search Engine
- Database - IBM DB2 7.2.3 or Oracle 8I DBMS 8.1.5
- IBM HTTP Server 1.3.6.3
- IBM WebSphere Application Server 3.5.3
- Java Development Kit 1.2
- JDBC drivers (as shipped with database product)
- Operating System – Sun Solaris, Windows NT, or AIX
- SFA Reusable Common Services 1.0
- SFA Portal Components Framework

### 3.2 Integration

Since the Portal Components were developed using the open source framework of Apache Struts and this framework implemented the Model-View-Controller (MVC) architecture, each of the Portal Components consists of three major parts: its business logic (the “model”), its presentation (the “view”), and its navigational flow (the “controller”). This architectural structure lends itself to being reusable in any of the following three ways:

Using the complete MVC package as it exists with no modifications. To illustrate, the Calendar Portal Component could be included on a web site to have the same appearance and use the same calendar data source as currently exists in the Schools Portal application.

Replacing the “model” with little to no modifications to the “view” and the “controller”. To illustrate, the Calendar Portal Component could be included on a web site to have the same appearance but use calendar data from a different data source than as currently exists in the Schools Portal application.

Replacing the “view” with little to no modifications to the “controller” and the “model”. To illustrate, the Calendar Portal Component could be included on a web site to use same calendar data source but have different appearance as currently exists in the Schools Portal application.

This architectural structure does not lend itself easily to replacing the “controller” since the majority of its functionality comes built within the Apache Struts framework.

### **3.3 Test Results**

All tests performed as part of the test plan were successfully executed utilizing a Schools Portal application with the Portal Components in the SFA development environment. For a specific listing of these tests, see the SFA Portal Strategy Tests spreadsheet in Appendix C.

## **4 Technical Recommendations**

This section outlines some of the technical recommendations to consider in the following phases of the SFA Portal Strategy including the Portal Components and the Schools Portal application. These recommendations are to create more modular, better performing, and easily maintainable software that is consistent with best practice principles.

### **4.1 Recommendations for Site Presentation**

Develop new or alter existing presentation requirements while remembering the benefit of commonality is the reduction in application development and maintenance costs. Specific examples to address include the subtle differences in title bar appearances and the multiple ways of sorting or organizing data such as in Headlines versus Calendar.

Utilize the existing set of Page Construction Tags provided by Apache Struts to create the user interface rather than creating from scratch with straight HTML. Specific examples to address include buttons and images.

Eliminate the usage of Java Script to improve performance by removing the client-side processing of this code. Specific examples to address include the header and search.

Migrate text strings into a properties or constants file to allow easier customization for reuse. Specific examples to address include the header and static help pages.

Replace numeric table sizes with percentages to improve the presentation for various screen resolutions. Specific examples to address include the header and static help pages.

## 4.2 Recommendations for Application Control

Modify the Interwoven SFA Links template for the Schools Portal application to utilize data deployment to populate the appropriate database table rather than the current process of open deployment to move a template created static HTML page.

Create an Oracle database table to map SFA Links for the Schools Portal application (and possibly searchable sites of other applications) to its corresponding Autonomy spider name.

Redevelop the Advanced Search functionality of the Search Portal Component to build the list of sites that can be searched from the new mapping database table. In addition, utilize this table to map selected site(s) to its appropriate Autonomy spider name(s) that will be passed as parameter(s) to the search engine to indicate where to search.

NOTE: Completing the aforementioned three steps eliminates the existing procedural step to alter the Schools Portal application software every time a new SFA link needs to be searchable. Instead, the process will be simplified to where SFA uses Interwoven to add the SFA link and notifies ITA to make the new site searchable and then the ITA makes the new site searchable and updates the mapping table with the information. Now, the site is searchable via the Advanced Search functionality of the Schools Portal application.

## 5 Architectural Recommendations

This section outlines some of the architectural recommendations to consider in the following phases of the SFA Portal Strategy.

### 5.1 Pros of Apache Struts

Apache Struts addresses and resolves the following issues by providing a framework that isolates and encapsulates each of the model, view, and controller components of a web application.

Apache Struts allows mapping HTTP parameters to JavaBeans. One of the most common tasks facing servlet programmers is to map a set of HTTP parameters (from the command line or from the POST of an HTML form) to a JavaBean for manipulation. This can be done using the `<jsp:useBean>` and `<jsp:setProperty>` tags, but this is cumbersome as it requires POSTing to a JSP, something that is not encouraged in a MVC architecture.

Apache Struts allows for validation. There is no standard way in servlet/JSP programming to validate that an HTML form is filled in correctly. This leaves every servlet programmer to develop his own validation procedures, or not, as is too often the case.

One of the more insidious problems in a servlet architecture is that URLs are usually coded directly into the code of the calling servlet in the form of a static string reference. This means that it is impossible to reorganize the JSPs in a Web site, or even change their names, without updating Java code in the servlets.

### 5.2 Cons of Apache Struts

Apache Struts lacks some functionality by being a new open source framework. However, as newer versions are released, the following additional functionality will be available.

Implementing the "Multibox" tag, which is described in the JavaDoc for 1.1 as follows: "Tag for input fields of type "checkbox". This differs from CheckboxTag because it assumes that the underlying property is an array getter (of any supported primitive type or String), and the checkbox is initialized to "checked" if the value listed for the "value" attribute is present in the values returned by the property getter."

Adding struts "templates" which are described as "tags that are useful in creating dynamic JSP templates for pages, which share a common format". This has been called the "Tiles" project and looks very useful for portal-style systems. As described in the JavaDoc for 1.1, "These templates are best used when it is likely that a layout shared by several pages in your application will change. The functionality provided by these tags is similar to what can be achieved using standard JSP include directive, but are dynamic rather than static."

Generating automatically client-side validation code (in JavaScript) by the struts taglibs. The idea is to hook together the server-side and client-side validation more tightly and perform the validation in the right place.

### 5.3 WebSphere Portal Server Considerations

A commercial off the shelf portal framework such as WebSphere Portal Server has several advantages including the ability to use commercial components that can dramatically reduce development costs and improve portal functionality. Commercial portals also provide component administration, personalization, customization, content syndication, and integrated search capabilities. The disadvantage is the initial cost of the portal software and the need to still do portal development if commercial portal components are not available.

## 6 Appendix A – File List

### 6.1 SQL database scripts

```
..\sfa\build\scripts\create_sfa_db.sql  
..\sfa\build\scripts\oracle_populate_sfa_db.sql  
..\sfa\build\scripts\populate_sfa_db.sql  
..\sfa\build\scripts\remove_sfa_db.sql
```

### 6.2 Java source

```
..\sfa\servlets\sfa\portlet\SFACalendarMapper.java  
..\sfa\servlets\sfa\portlet\CheckLogonTag.java  
..\sfa\servlets\sfa\portlet\DateFormatter.java  
..\sfa\servlets\sfa\portlet\DisplayHeaderAction.java  
..\sfa\servlets\sfa\portlet\DisplayHeadlinesAction.java  
..\sfa\servlets\sfa\portlet\DisplayLinksAction.java  
..\sfa\servlets\sfa\portlet\DisplayLogonAction.java  
..\sfa\servlets\sfa\portlet\DisplayLogonTitleBarAction.java  
..\sfa\servlets\sfa\portlet\DisplayTestDBAction.java  
..\sfa\servlets\sfa\portlet>EditBookmarkAction.java  
..\sfa\servlets\sfa\portlet>EditLinkAction.java
```

..\sfa\servlets\sfa\portlet\EditRegistrationAction.java  
..\sfa\servlets\sfa\portlet\LinkForm.java  
..\sfa\servlets\sfa\portlet\LinkLinkTag.java  
..\sfa\servlets\sfa\portlet\LinkUserTag.java  
..\sfa\servlets\sfa\portlet\LogoffAction.java  
..\sfa\servlets\sfa\portlet\LogonAction.java  
..\sfa\servlets\sfa\portlet\LogonForm.java  
..\sfa\servlets\sfa\portlet\ProcessCalendarAction.java  
..\sfa\servlets\sfa\portlet\RegistrationForm.java  
..\sfa\servlets\sfa\portlet\SaveLinkAction.java  
..\sfa\servlets\sfa\portlet\SaveRegistrationAction.java  
..\sfa\servlets\sfa\portlet\SFABookmarksBO.java  
..\sfa\servlets\sfa\portlet\SFABookmarksBOFactory.java  
..\sfa\servlets\sfa\portlet\SFACalendar.java  
..\sfa\servlets\sfa\portlet\SFACalendarBO.java  
..\sfa\servlets\sfa\portlet\SFACalendarBOFactory.java  
..\sfa\servlets\sfa\portlet\SFACalendarEmitter.java  
..\sfa\servlets\sfa\portlet\CalendarForm.java  
..\sfa\servlets\sfa\portlet\SFACalendarView.java  
..\sfa\servlets\sfa\portlet\SFACalendarViewBeanProcessor.java  
..\sfa\servlets\sfa\portlet\SFAHeadlineBO.java  
..\sfa\servlets\sfa\portlet\SFAHeadlineBOFactory.java  
..\sfa\servlets\sfa\portlet\SFAHeadlineMapper.java  
..\sfa\servlets\sfa\portlet\SFALinkBO.java  
..\sfa\servlets\sfa\portlet\SFALinksBO.java  
..\sfa\servlets\sfa\portlet\SFAPortletConstants.java  
..\sfa\servlets\sfa\portlet\SFAPortletMethods.java  
..\sfa\servlets\sfa\portlet\SFASystemLinksBO.java  
..\sfa\servlets\sfa\portlet\SFASystemLinksBOFactory.java  
..\sfa\servlets\sfa\portlet\SFAUserBO.java  
..\sfa\servlets\sfa\portlet\SFAUserBOFactory.java  
..\sfa\servlets\sfa\portlet\TagViewbeanProcessor.java  
..\sfa\servlets\sfa\portlet\TagViewbeanProcessorFactory.java  
..\sfa\servlets\sfa\portlet\ViewbeanTag.java

### 6.3 HTMLs

..\sfa\sfa\web\blank.html  
..\sfa\web\bookmark\_help.html  
..\sfa\web\calendar\_help.html  
..\sfa\web\frame\_border.html  
..\sfa\web\frame\_bottom.html  
..\sfa\web\header.html  
..\sfa\web\headermy.html  
..\sfa\web\index.html  
..\sfa\web\login\_help.html

..\sfa\web\parts\_help.html  
..\sfa\web\password\_help.html  
..\sfa\web\portal\_contacts.html  
..\sfa\web\portal\_customize.html  
..\sfa\web\portal\_faq.html  
..\sfa\web\portal\_feedback.html  
..\sfa\web\portal\_gaquestion.html  
..\sfa\web\portal\_getbrowsers.html  
..\sfa\web\portal\_help.html  
..\sfa\web\privacy.html  
..\sfa\web\search.html  
..\sfa\web\search\_help.html  
..\sfa\web\sfallink\_help.html  
..\sfa\web\basic\_search\_help.html

#### 6.4 JSPs

..\sfa\web\advanced\_search.jsp  
..\sfa\web\bookmarksedit.jsp  
..\sfa\web\calendar.jsp  
..\sfa\web\calendar\_categories.jsp  
..\sfa\web\calendar\_titlebar.jsp  
..\sfa\web\changePASS.jsp  
..\sfa\web\header.jsp  
..\sfa\web\headermy.jsp  
..\sfa\web\headlines.jsp  
..\sfa\web\headlines\_archive.jsp  
..\sfa\web\headlines\_titlebar.jsp  
..\sfa\web\index.jsp  
..\sfa\web\linkedit.jsp  
..\sfa\web\links.jsp  
..\sfa\web\links\_titlebar.jsp  
..\sfa\web\linksedit.jsp  
..\sfa\web\linksmy.jsp  
..\sfa\web\logon.jsp  
..\sfa\web\logon\_titlebar.jsp  
..\sfa\web\personalize.jsp  
..\sfa\web\registration.jsp  
..\sfa\web\registration\_confirmation.jsp  
..\sfa\web\registration\_confirmtable.jsp  
..\sfa\web\search\_titlebar.jsp  
..\sfa\web\searchadvanced.jsp  
..\sfa\web\searchresults.jsp  
..\sfa\web\searchsuggest.jsp  
..\sfa\web\master\_frameset.jsp

## 6.5 GIFs

..\sfa\web\images\SFAnetscape.gif  
..\sfa\web\images\addlink2.gif  
..\sfa\web\images\bookmark\_help\_step1.gif  
..\sfa\web\images\bookmark\_help\_step2.gif  
..\sfa\web\images\bookmark\_help\_step3.gif  
..\sfa\web\images\bottompic.gif  
..\sfa\web\images\but\_change\_pass.gif  
..\sfa\web\images\but\_submit1.gif  
..\sfa\web\images\but\_update.gif  
..\sfa\web\images\corner-left.gif  
..\sfa\web\images\corner-right.gif  
..\sfa\web\images\frame\_bg.gif  
..\sfa\web\images\hedr-logo.gif  
..\sfa\web\images\hedr-title.gif  
..\sfa\web\images\hedr-title2.gif  
..\sfa\web\images\home\_sm.gif  
..\sfa\web\images\link\_bookmarks1.gif  
..\sfa\web\images\link\_sfalinks1.gif  
..\sfa\web\images\linkbar\_end1.gif  
..\sfa\web\images\linkbar\_end2.gif  
..\sfa\web\images\map\_usa.gif  
..\sfa\web\images\parts\_help\_calendar.gif  
..\sfa\web\images\parts\_help\_header.gif  
..\sfa\web\images\parts\_help\_headlines.gif  
..\sfa\web\images\parts\_help\_login.gif  
..\sfa\web\images\parts\_help\_password.gif  
..\sfa\web\images\parts\_help\_search.gif  
..\sfa\web\images\parts\_help\_sfalinks.gif  
..\sfa\web\images\SFAdeselect.gif  
..\sfa\web\images\SFAget\_ie.gif  
..\sfa\web\images\sfallink\_help\_step1.gif  
..\sfa\web\images\sfallink\_help\_step2.gif  
..\sfa\web\images\addlink.gif  
..\sfa\web\images\SFAselect.gif  
..\sfa\web\images\slogan.gif  
..\sfa\web\images\spacer.gif  
..\sfa\web\images\toplink\_contacts.gif  
..\sfa\web\images\toplink\_faqs.gif  
..\sfa\web\images\toplink\_feedback.gif  
..\sfa\web\images\toplink\_home.gif  
..\sfa\web\images\toplink\_mysfa.gif  
..\sfa\web\images\toplink\_portalhelp.gif  
..\sfa\web\images\toplink\_question.gif  
..\sfa\web\images\toplink\_support.gif

..\sfa\web\images\SFAadvSearch.gif  
..\sfa\web\images\SFAsearchhelp.gif  
..\sfa\web\search\_files\rightshadow.gif  
..\sfa\web\search\_files\blue\_corner.gif  
..\sfa\web\search\_files\doe\_seal.gif  
..\sfa\web\search\_files\icon\_csb.gif  
..\sfa\web\search\_files\ifap\_logo.gif  
..\sfa\web\search\_files\pixel.gif  
..\sfa\web\search\_files\background.gif  
..\sfa\web\search\_files\search\_b.gif  
..\sfa\web\search\_files\title\_end.gif  
..\sfa\web\search\_files\top\_shadowbar\_r.gif  
..\sfa\web\search\_files\top\_shadowbar\_r2.gif  
..\sfa\web\search\_files\top\_shadowbarleft.gif  
..\sfa\web\search\_files\yellow\_background.gif

## **6.6 Configuration Files**

..\sfa\web\sfa.webapp  
..\sfa\web\search\_files\Master.css  
..\sfa\web\search\_files\ifap.css  
..\sfa\web\WEB-INF\search.tld  
..\sfa\web\WEB-INF\web-app\_2\_2.dtd  
..\sfa\web\WEB-INF\web-app\_2\_3.dtd  
..\sfa\web\WEB-INF\ApplicationResources.properties  
..\sfa\web\WEB-INF\app.tld  
..\sfa\web\WEB-INF\struts-config\_1\_0.dtd  
..\sfa\web\WEB-INF\sfaportal-sfa.tld  
..\sfa\web\WEB-INF\struts.tld  
..\sfa\web\WEB-INF\struts-bean.tld  
..\sfa\web\WEB-INF\struts-form.tld  
..\sfa\web\WEB-INF\struts-html.tld  
..\sfa\web\WEB-INF\struts-logic.tld  
..\sfa\web\WEB-INF\struts-template.tld  
..\sfa\web\WEB-INF\database.xml  
..\sfa\web\WEB-INF\struts-config.xml  
..\sfa\web\WEB-INF\web.xml

## **7 Appendix B – Installation Instructions**

Please see SFA Portal Strategy Install Specification Word document (TO48\_Portal\_Strategy\_Install\_Spec.doc).

## **8 Appendix C – Test Plan**

Please see SFA Portal Strategy Test Plan Excel document (TO48\_Portal\_Strategy\_Test\_Plan.xls).

