empowerNET

Portal Functional Specification

Version 0.21

May 2001
Convergent Group, U0352

This is a privileged and confidential document between Cinergy and Convergent Group. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), without prior written consent from Cinergy and Convergent Group.

Revision History

Date	Version	Description	Author
05/22/01	0.1	1st Draft	Brian Loomis
6/20/01	0.2	Added login process and security info	Brett Stallman
6/25/01	0.2	Added info on event logging component for login, app launch, and logout. Also added info on the application launch process.	Brett Stallman
6/27/01	0.21	Added detailed info on log component	Brett Stallman

ToDo List

1. Add screen snapshots – brianl
2. Fully define user object – brianl
3. Fully define Citrix and SM components – bretts
4. Fully define admin and AD/application data services components -- mattm

5. Table of Contents

Introduction

EmpowerNET is an Applications Service Provider (ASP) responsible for hosting complex operational systems to utilities and energy service providers via an application rental schedule. As an ASP, EmpowerNET will offer key fully integrated systems that service a large number of the business functions required to operate an energy company.

EmpowerNET is being developed using a rapid application development methodology comprised of 1-2 week software development iterations leading up to a beta release of the software at a milestone named Generation 1.

1. Purpose

The purpose of this document is to provide a functional specification of the software elements making up the portal component of the EmpowerNET offering. The document provides detailed user scenarios, an overview of the technical software architecture, and specific functional specifications for each subsystem included in Generation 1. The document is to be used to test and validate that the Generation 1 portal meets known requirements and can be used as a baseline for subsequent generations.

2. Overview

The document contains the following key sections

Feature identification – A list of the feature sets and their designators for all portal development

Product definition – A summary description of the portal product and its relationship to other EmpowerNET services

User scenarios – A description of the actors who interact with the portal system and a master list of use cases for the portal

Architecture discussion – A description of the overall technical architecture and notes to accompany the architecture diagrams (in VISIO)

Specific functional specifications (by feature set) – For each feature set, this describes the specific presentation, business, and data services required to meet the use cases. This includes screen snapshots, pseudo-code, and database schema design as necessary.

3. Other Documents

The following documents support this functional specification[1]:

- Active Directory.VSD – a physical layout of Active Directory elements
- Database schema.VSD – a logical and physical layout of database elements
- Repository_G1.sql – final scripts to generate the Generation 1 database
- Portal.VSD – a web site diagram
- Portal Test Plan.DOC

4. Product Definition

The EmpowerNET portal application provides a secure, web-based, graphical experience for EmpowerNET customers, typically utility customers and their customers, to access critical, hosted services. The portal provides static content regarding EmpowerNet, a variety of dynamic content for a superior portal experience (weather, messaging, industry news, etc.), and hosted applications (through Citrix terminal services or directly through Enterprise Application Integration pages). The portal finally provides a web-based administrative interface to manage services, customer accounts, and security settings.

The EmpowerNET portal requirements and requirements for other applications are detailed in the Software Requirements Specification and the Software Architecture Document.

5. User Scenarios

The use cases for the system are defined in separate documents but are summarized here for reference. The use cases refer to specific roles ("actors") and external system interaction points ("systems"). Actors include administrators and end/support users for each of the following groups: EmpowerNet, utility customer, commercial and industrial (C & I) customer; also there is an anonymous/guest user. Systems include billing, metering, load/supply management, MWM/CAD, AM/FM/GIS, OMS, DPS, WMS, PA, SCADA, CIS, and IVR systems.

This list contains scenarios for iteration 1 only.

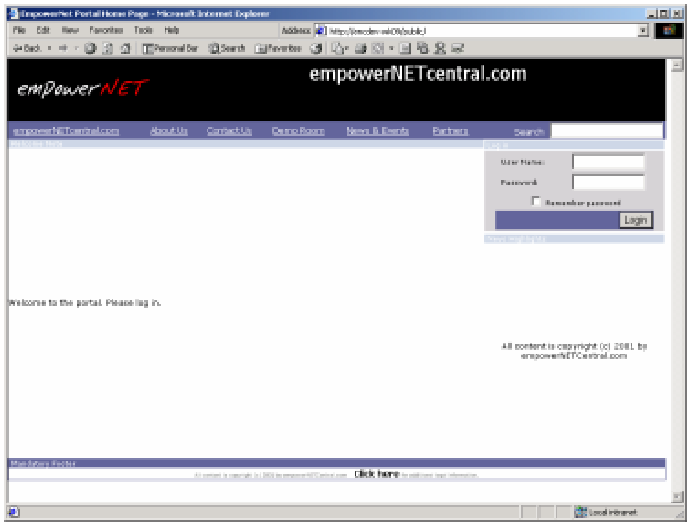
5.1 Security scenarios

5.1.1 End-user portal

5.1.1.1 Unauthenticated (anonymous) access

Default.aspx is the standard starting point for unauthenticated users. Additional scenarios for this user role are listed in static content section below.

[1] All files are in the G1/Development\Team Portal\Functional Specification folder



Welcome to the portal. Please log in.

All content is copyright (c) 2001 by empowerNETCentral.com

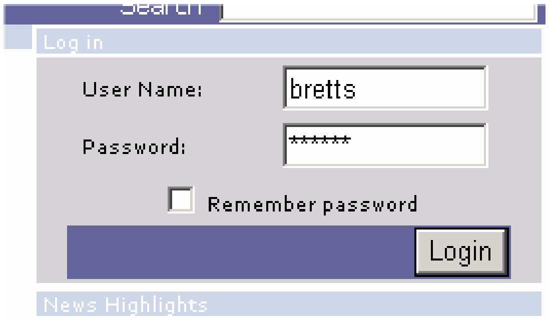
Figure 1 - Anonymous default page

5.1.1.1.1 Login

Unauthenticated users will have a form (control) to log in to the authenticated part of the site.

The control is defined by Login.ascx, which loads with a target URI configured (passthru and target explained below) via its tag declaration in default.aspx.

The unauthenticated user enters his credentials into the login form (consisting of username and password). The user then hits the Login button.



User Name: bretts

Password: ******

Figure 2 - Login Control

Server posts the username and password fields and fires the btnLogin event handler for Login.ascx. The server handling the login method attempts to make an internal request on the user’s behalf (call this the IsAuthenticated attempt) using the credentials supplied and the target URI to determine if the target URI will accept the credentials (or if the target resource is even available).

Currently, three outcomes are defined for the IsAuthenticated attempt:

- The IsAuthenticated attempt succeeds with status of OK.
- The IsAuthenticated attempt returns a status of Forbidden.
- The IsAuthenticated attempt returns a Server Internal Error (500), or some other non-forbidden, but equally bad error.

If the error is forbidden, the label below the user's login form changes to "Authentication Failed".

If there is some other error, by which the internal request did not go through, the label below the user's login form changes to "Server error, can't continue"

On every failed login attempt (caused by either Forbidden or other error), the logging component (Empowernet.Web.Utils.Log) records the attempt with the credentials that were used in the attempt.

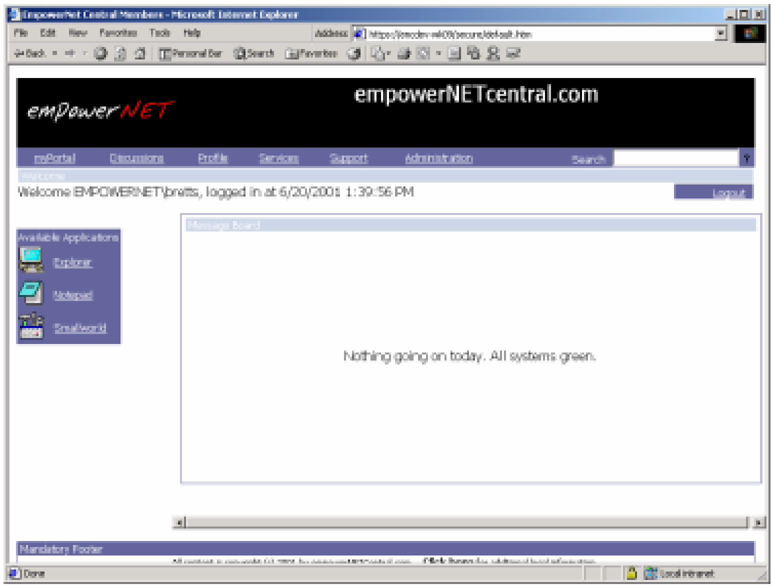
If the IsAuthenticated attempt succeeds the following events occur:

- The label message is briefly changed to "Authenticated"
- The logging component records a successful login event
- 3 cookie values are set for NFuse interop (Userid, Password, and Domain),
- The user is forwarded to the target URI. Currently, the target URI must be hardcoded int the tag declaration like "https://someserver/secure/passthru.aspx?target=default.htm")

Note: passthru.aspx resides under the Secure portion of the portal. It has inline server code which redirects the user to the target specified after the question mark in the URI. If we were to go directly to default.htm instead of using passthru.aspx, the user would see his/her credentials in the address bar of IE (**http://user:password@someserver.empowernetcentral.com/default.htm**).

Note: The target URI **MUST NOT** contain server side render tags (<% or <%=) as these are not processed until AFTER the server side code (event handlers) execute, which will handle IsAuthenticated attempts and eventual redirection. Also, the target URI **SHOULD NOT** point to any resource that is not accessible by external users.

If the user navigates directly to the secure portion of the site, Basic authentication will prompt the user to enter his/her credentials. This login will **not** be recorded in the database. Also, the user will not be able see what Citrix applications are available for him/her since they did not use the login server control.



Welcome EMPOWERNET\bretts, logged in at 6/20/2001 1:39:56 PM

Nothing going on today. All systems green.

Figure 3 - Authenticated User Default Page

5.1.1.1.2 Forgot user-id request

Send support phone number to user's email address.

5.1.1.1.3 Forgot password request

Send hint notification and support phone number to user's email address.

5.1.1.2 Login (user in Utility or C+I end users group)

Users in this group will see the application hosting menu based on their authorization (see application hosting below).

5.1.1.2.1 Change password

5.1.1.3 Login (user in EmpowerNet administrative or support, Utility, or C+I administrators group)

All scenarios in Utility end user group plus the following:

5.1.1.3.1 Administrative menu

Members of the admin groups will get an additional menu item providing access to the administrative portal.

5.1.2 Administrative portal

5.1.2.1 Unauthenticated (anonymous) access

Access will be denied to users in this category (or if authentication fails), and the user will be redirected to a failed login page.

5.1.2.2 Authenticated Utility or C+I end-user login

Access will be denied to users in this category, and the user will be redirected to an unauthorized login page (must be administrator).

5.1.2.3 Login (authenticated user in EmpowerNet administrators group)

5.1.2.3.1 Add a new service type

5.1.2.3.2 Edit an existing service type

5.1.2.3.3 Delete an existing service type

Check to see that it is not in use. Prompt to force deletion.

5.1.2.3.4 Add new utility (primary) or C+I/operating location (secondary) customer

Add utility OU or select OU to put new group (sub-OU) under. Create a new associated plan and select services and restrictions on services (max number of concurrent users, etc.); if this is a new sub-OU, plan is defined only based on parent utility OU derived from. Add entry in application log table with transaction record and date stamp.

5.1.2.3.5 Edit utility (primary) or C+I/operating location (secondary) customer plan or details

Add or delete services from an associated (named) plan, modify restrictions on services or default profile/configuration for a selected plan, change customer details (name, mapped OU), add or delete an associated plan (name of plan, contact info for plan: phone, fax, userid, user name, billing address). Modify SiteMinder and NTFS credentials on directories. Add entry in application log table with transaction record and date stamp. A primary customer has one master set of services and one or more plans may be created from this set of services; each master set of services has overall restrictions; each individual plan has a list of services and their configuration details and can be named after an operating location. If we change configuration parameters, check a box to propagate settings to all users in affected group(s).

5.1.2.3.6 Delete utility (primary) or C+I/operating location (secondary) customer

Delete plan and OU and/or sub-OUs. Add entry in application log table with transaction record and date stamp.

5.1.2.3.7 Add new user

Specify user name (AD short name and long name), password, type of user (admin or not), default OU (Empowernet, utility OU, or C+I OU), set allowed services (based on default OU), set service profile settings (based on default OU), and email address. Also, specify if an email notification goes out to the user or not. Add entry in application log table with transaction record and date stamp.

5.1.2.3.8 Edit user profile or details

Change user name, password, type of user, allowed services, profile settings, or email address.

5.1.2.3.9 Delete registered user

Specify whether an email notification goes out or not to the user. Add entry in application log table with transaction record and date stamp.

5.1.2.3.10 Extract transaction log information

Search and select particular transaction log records (as XML). E.g., billing information for a specific utility, actions taken by a specific user, actions within a certain date range.

5.1.2.3.11 Publish new general message

Customer-specific messages will be in a later iteration.

5.1.2.3.12 Rescind existing general message

5.1.2.3.13 Export profile for specified user

5.1.2.3.14 Import profile for specified user

5.1.2.3.15 Submit support request

5.1.2.3.16 View/search support request queue

5.1.2.3.17 Respond to support requests (close out/retire the request and log it)

5.1.2.4 Login (user in EmpowerNet support users group)

Can do all operations of EmpowerNet administrator **except** add/modify/delete service types, add/modify/delete EmpowerNet users (admin or otherwise), change EmpowerNet user/admin groups,

5.1.2.5 Login (user in Utility administrators group)

Can do all operations of EmpowerNet support users group **except** add/modify/delete primary customers (including myself), add/modify/delete secondary customers (except those that I own), import or export user profiles, or respond to support requests. Additionally, utility administrators can do the following:

5.1.2.5.1 View plan or service information

5.1.2.5.2 Request change of plan or service

Request add/delete plan, add/delete services, change restrictions on services, change OU configuration settings. Request change to customer information (name, etc.).

5.1.2.5.3 Request termination of service

5.2 Static portal content scenarios (unsecured)

All scenarios in this section require only anonymous access to the portal.

5.2.1 Static content scenarios

5.2.1.1 About us (company overview, products, subscribe to newsletter, partners, jobs, US/Worldwide sites), Contact Us (product support, sales support)

5.2.1.2 Weather, stock ticker

5.2.1.3 Search site

5.2.1.4 Industry news feed

5.2.1.5 Empowernet message display (press news, events)

5.2.1.6 Request a demo account

5.3 Application hosting scenarios (secure)

All scenarios in this section require authenticated access to the portal as an Empowernet user plus service plan definition permitting access.

5.3.1 Selection of authorized applications

5.3.1.1 Determine authorized hosted applications for an authenticated user (menu)

5.3.2 Citrix hosting scenarios

Once a user has been authenticated and their NFUSE cookies have been set, a server control will list the applications available (via Empowernet.Secure.Controls.NFuse) to that user. If the cookies are not set the server control will display “`Profile information has not been provided. Please logout and log back in.`”.

Each of these applications will be displayed as links that point to **prelaunch.aspx** with particular querystring attributes set.

The purpose of **Prelaunch.aspx** is to log the applaunch event (via Empowernet.Web.Utils) for the particular app/user. It then redirects the user to **launch.aspx** which does the actual work of sending an ICA file down to the user describing to the installed ICA client handler how to launch the Citrix application.

5.3.2.1 Launch Small World

5.3.2.2 Launch Silicon Energy

5.3.2.3 Launch CES

5.4 Portal EAI scenarios

5.4.1 Test (non-deployed) scenarios

5.4.1.1 MSMQ message submission form

6. Architecture Discussion

Conceptual, or logical, business objects are captured through requirements analysis and are refined into designed, or physical, classes. Typically a single logical business object will have a corresponding class in each of the presentation services, business services, and data services tiers at implementation. After the logical object model is drafted, separate diagrams plus interaction diagrams for each tier are designed and will be maintained in the project design folder. All objects will be grouped into libraries (DLLs) or assemblies after considering performance implications, network deployment, and other architectural aspects.

For logical design documents of the portal application in Generation 1, please refer to the Visio diagrams in the project documentation folder. The database schema diagram is also contained in a separate Visio diagram.

The sections below describe the functional specification (design) of the user/presentation services, business services, and data services.

6.1 Presentation services overview

The functional specification for this section will describe the active server pages, dashboard parts, global application settings, web services interfaces. This section will include:

- Application and session state (and configuration files)
- Default portal page
- Default administration page
- Additional pages supporting use cases above
- Web services interface specification
- Screen mockup snapshots
- Data validation for inputs
- Script code for interaction with business services

6.2 Business services overview

The functional specification for this section will describe the business logic components. This section will include:

- Administrative operations component (managing plans, user, groups)
- User operations component (profile persistence, import/export, search, request queue)

6.3 Data services overview

The functional specification for this section will describe the components for accessing different repositories and the application repository schema. This section will include:

- Active Directory services component
- SiteMinder services component
- Application data services (and logging) component

The logging component must be initialized with 4 parameters matching standard SQL database login information. Currently, to create an instance of the Empowernet.Web.Utils.Log class you declare something like:

```
   Log appLog = new Log("dbserver1", "mydatabase", "sa",
"sapassword");
```

The first parameter describes the server the database resides on, the second is the name of the database (catalog) to which you are connecting, the last two parameters are user credentials used to access the database (UserID and Password respectively).

To ensure these parameters are not hard-coded into every constructor, we will use a static class residing under global.asax.cs to define 4 static variables which will contain the values for the database parameters. This allows us to declare the Log class as follows with the LogSettings class having 4 static-public-read-only member variables as defined in global.asax.cs :

```
Log appLog = new Log(LogSettings.Server, LogSettings.Catalog,
LogSettings.UserID, LogSettings.Password);
```

The Log class resides inside Empowernet.Web.Utils namespace (in the Empowernet.Web.Utils.dll). The Utils namespace defines 3 Enums as well as the Log class.

Enums defined under Utils namespace:

AdminServiceEvents – define events for administrators operating on/with service types or service plans (AddNewServiceType, CreateNewServicePlan, etc.)

AdminUserEvents – define events for administrators managing users and groups (CreateUser, DeleteUser, etc.)

AdminUtilityEvents – define events for administrators managing Utilities (Add,Delete,Edit Utility)

LoginGroup – defines the two groups of users who require authentication: Administrators and Secure Users.

The Log class defines the following Methods:

AdminServiceEvent(AdminServiceEvents x, …) – record events for ServiceEvents. Specify which event occurred using the Enum class as the first argument.

AdminUserEvent(AdminUserEvents x…) - record events for UserEvents. Specify which event occurred using the Enum class as the first argument.

AdminUtilityEvent(AdminUtilityEvents x…) - record events for UtilityEvents. Specify which event occurred using the Enum class as the first argument.

AppLaunch(LoginGroup x…) - record Application launches. Specify which group of users this request is on behalf of using the LoginGroup enum as the first argument.

FailedLogin(string, string) – record failed logins using description and machine name (The caller should specify what the attempted credentials were as part of the description).

Login(LoginGroup x…) – record successful logins. Categorize the event based on what group the user belongs to (Admin or Secure User).

Logout(LoginGroup x) – record logouts

PasswordChange(LoginGroup x) – record password change events

- MSMQ services component
- Citrix services component
- Application repository schema discussion

6.4 Deployment considerations

6.4.1 Web Parts, Digital Dashboard, ASP.NET

Digital dashboards are composed of a virtual directory on an IIS server, IIS meta-base entries (SQL Server connection attributes), and a SQL Server repository (DAV catalog). The virtual directory can be copied and reinstalled on new machines (e.g., across a production server farm). The IIS meta-base settings can be modified through the DDRK 3.0 install program (following the selection to modify installation parameters). The SQL repository can be simple backed up and restored on the target server. Additional information on deployment exists in the DDRK 3.0 help file.

ASP.NET requires Windows Component Update on web server machines.

6.4.2 Citrix NFuse, MetaFrame

6.4.3 SQL Server

7. Portal Functional Specification (user services)

This section will describe the active server pages, dashboard parts, global application settings, web services interfaces. This section will include:

- Application and session state (and configuration files)
- Default portal page
- Default administration page
- Additional pages supporting use cases above
- Web services interface specification
- Screen mockup snapshots
- Data validation for inputs
- Script code for interaction with business services

The web site map details physical location for each of the files mentioned herein (see web site map in Visio diagrams).

The portal initializes itself from a config.web file in the root directory of each site; these values can be loaded from individual web controls as application state values. The schema for the public configuration file (http://www.empowernetcentral.com/public/config.web) is:

```
<xml>
      <welcomeFile>public/content/foo.htm</welcomeFile>
      <newsHighlightsFile>public/content/highlights.xml</newsHighlightsFile>
      <newsDirectory>public/content/news</newsDirectory>
      <aboutUsFile>public/content/aboutus.xml</aboutUsFile>
      <aboutUsDirectory>public/content/about</aboutUsDirectory>
      <partnerFile>public/content/partners.xml</partnerFile>
      <partnerDirectory>public/content/partner</partnerDirectory>
      <contactEmail>sales@empowernetcentral.com</contactEmail>
</xml>

‘ news highlights schema
<newsHighlights>
      <newsHighlight title=”EmpowerNet Rocks!” file=”rocks.htm”/>
      <newsHighlight title=”…” file=”…”/>       ‘ files in news directory
</newsHighlights>

‘ news article schema
<article sequence=”1”>
      <title>Lights on at EmpowerNet</title>
      <text>For the first time, … </text>
```

```
</article>
```

The schema for the secure configuration file (http://www.empowernetcentral.com/secure/config.web) is:

```
<xml>
	<ENCadmingroup>someActiveDirectoryOU</ENCadmingroup>
	<ENCusersgroup>someActiveDirectoryOU</ENCusersgroup>
	<MessagesUNCDirectory>secure/messages</MessagesUNCDirectory>
	<repositoryString>server=foo;database=bar;user=sa;pwd=</repositoryString>
</xml>
```

7.1 Extranet (not secure)

The public site will consist of a default page (http://www.empowernetcentral.com/public/default.aspx) which presents certain static content (using web controls) and a button to log in to the secure application hosting site. No business components are used on the extranet site. The static content will consist of the following web controls:

- Logo_banner (displays EmpowerNet logo and text banner for all pages)
- Public_main_menu (displays menu items for empowerNETCentral.com (link to `default.aspx`), About Us, Contact Us, Partners, News and Events, and Demo Room; each menu item links to a different page with web controls; the default page's web controls are a public_text_displayer (with welcome message), public_login, mandatory_footer and public_news_highlights web control.
- Public_login (displays a button to go to SiteMinder login page)
- Public_text_displayer (responds to events defining the web control's title and file path; when file path is set, the web control loads the requested file in as innerHTML)
- Public_aboutus_menu (displays a list of menu choices from the application configuration file and loads the requested HTML file into a public_text_displayer web control when clicked)
- Public_partner_menu (reads partner list from application configuration file and dynamically creates partner site menu; partner links are rendered in new windows other than the current browser window)
- Public_news_highlights (reads highlight list from application configuration and displays them as links to the news_menu page)
- Public_news_menu (reads news and events article summaries from a file specified in the application configuration file and creates links to a public_text_displayer control for each article)
- Public_contactus_menu
- Public_demo_room (displays a mailto link to contact EmpowerNet, displays a button to sign up (populates text displayer with a form mailed to ENC for account creation), button to view demo (loads Flash player in text displayer))
- Mandatory_footer (displays copyright information and link to legal disclaimer page)

The other pages in the default site are (with web components in parentheses after):

- `Aboutus.aspx` (logo_banner, public_main_menu, public_aboutus_menu, public_text_displayer, public_login, mandatory_footer)
- `Contacts.aspx` (logo_banner, public_main_menu, public_contactus_menu, public_text_displayer, public_login, mandatory_footer)

- `Partners.aspx` (logo_banner, public_main_menu, public_partner_menu, public_text_displayer, public_login, mandatory_footer)

- `Demo.aspx` (logo_banner, public_main_menu, public_demo_room, public_text_displayer, public_login, mandatory_footer)

- `News.aspx` (logo_banner, public_main_menu, public_news_menu, public_text_displayer, public_login, mandatory_footer)

7.2 Secure application hosting

The login button from the extranet (public) site and any unauthenticated page reference to the secure site will redirect the user to the SiteMinder login page (http://www.empowernetcentral.com/secure/sm.fcc) to enter user credentials within an HTTPS session. If credential authentication succeeds, the user will be forwarded to either the requested page or, if no specific, secured page was requested, to http://www.empowernetcentral.com/secure/default.aspx (SiteMinder also fills in HTTP variables indicating authentication succeeded). The secure application hosting site will consist of the following web controls:

- Logo_banner (as above)
- Secure_main_menu (detects if the user is an administrator and adds a link to the admin page, adds a menu item for change password)
- Mandatory_header (displays authenticated user name, search control, and logout button): Logout button destroys session state and cookies and returns the user to the public default page.
- Secure_application_hosting_menu (reads the user profile from the user component and dynamically builds hyperlinks to Citrix sessions available to the user; automates the popup NFuse windows and passing of credentials to start these sessions)
- Secure_message_board (reads messages of the day from directory pointed to in application configuration file)
- Mandatory_footer (as above)

The secure application hosting site will use the user business object (loaded into session state on login) to get profile information based on user credentials composed from the SiteMinder HTTP headers (SM_SDOMAIN, SM_SUSER, and SM_AUTHORIZED). The secure site will also check if the user is in the administrator role for the authenticated group (or OU); if the user is an administrator, an administrator business object will be loaded into session state.

Change password will be on a separate page (http://www.empowernetcentral.com/secure/pwd.aspx) and will have the following web control in addition to the ones specified above (and without the secure hosting application menu control):

- Secure_change_password (allows user to select a user id and group to change, button to change password when given old password, new password and confirmed new password); to change users other than myself, the user must be an admin of the OU (or immediate parent OU) or EmpowerNet support group

The following schema defines the services DOM for a particular user (stored in AD profile attribute for the user):

```
<xml>
      <plan>operations 1</plan>
      <services>
            <service type=”smlwrld” name=”inst1” display=”Link to my SmallWorld”
                        href=”www.citrix.com/foo.htm”>
                  <parameters>
                        <parameter tag=”myconfigsetting1” value=”someval”/>
                        <parameter tag=”mydatabase” value=”sql2000”/>
                  </parameters>
            </service>
            <service type=”ces” name=”inst2” display=”Link to my CES”
                  href=”www.citrix.com/bar.htm”>
            </service>
      </services>
</xml>
```

7.3 Administration (secure)

The administration page (http://www.empowernetcentral.com/secure/admin.aspx) uses the following web controls: logo_banner, secure_main_menu, mandatory_header, secure_admin_ops_menu, and mandatory_footer. Most of these pages will be based on the scripting of the Microsoft Web Administration Tool, but the logic will be componentized and placed in the AD services component rather than reside in script.

The following is the complete list of specific administrative web controls used on all admin site pages (components used by the web control are listed afterwards in bold text and parentheses: **(SQL)** indicates the control uses functions from the Application data services component, **(AD)** indicates that the control uses the AD services component[2]):

- Secure_main_menu (as above)
- Logo_banner (as above)
- Mandatory_header (as above)
- Mandatory_footer (as above)
- Secure_admin_ops_menu (reads the admin profile from the admin component and dynamically builds hyperlinks to the specific admin pages as available to the specific admin ops component and user business component in session state)
- Secure_admin_service_types (lists **all** service types, buttons for add/edit/delete service type; add/edit populate a details panel with service type name, description, and configuration spec which can be pasted or loaded from file with browse button; delete prompts administrator to force deletion of both service type and instances if a service instance exists for the type; as logged events) **(SQL)**
- Secure_admin_select_SLA_for_edit (lists **all** SLAs, buttons to add/modify/delete, Delete as logged event) **(SQL)**
- Secure_admin_select_SLA_no_edit (lists **only** SLAs within administrator permissions) **(SQL, AD)**
- Secure_admin_edit_SLA (lists details of SLA (name, description, restrictions) and list box showing plans (no edits), buttons to submit) as logged event **(SQL)**
- Secure_admin_select_OU_for_edit (tree view to list/select OU (from the total AD, only show those available to the administrator, current plus any sub-OUs), buttons from add/modify/delete OU, Delete as a logged event) **(SQL, AD)**
- Secure_admin_select_OU_no_edit (lists all OUs available to administrator in a tree view, buttons for Add/Delete/Edit/Users, Delete as a logged event) **(SQL, AD)**

[2] via the administrative operations component

- Secure_admin_edit_OU (lists details of selected OU (name, contact info, etc.), list box to select SLA (only for EmpowerNet users), list box showing service plan(s), button to submit/save changes) as logged event **(SQL, AD)**

- Secure_admin_select_user_for_edit (lists all users in selected OU, buttons for Add/Delete/Edit, Delete is a logged event) **(SQL, AD)**

- Secure_admin_edit_user (entry fields for AD user name (not changeable), long name, password, admin checkbox, service profile as checkboxes (“inherit all” from group plan or specify a plan from the list of plans for the group, ENC users can import/export to XML file) and text configuration settings (for the services in the selected plan), email address) as logged event **(SQL, AD)**

- Secure_admin_new_user (entry fields for AD user name, long name, password, admin checkbox, service profile as checkboxes (“inherit all” from group plan or specify a plan from the list of plans for the group) and text configuration settings (for the services in the selected plan), email address) as logged event **(SQL, AD)**

- Secure_admin_select_plans_for_edit (lists plans within selected group, buttons to add/modify/delete, Delete as logged event, prompt that this will propagate changes to affected users) **(SQL, AD)**

- Secure_admin_select_plans_no_edit (lists plans within selected group and SLA) **(SQL)**

- Secure_admin_edit_plan as logged event (lists details of selected plan (name, description) and list boxes displaying selected Service Instances for this plan and ones available within the SLA, button to submit and add service to Plan from SLA list, remove service, prompt that this will propagate changes to affected users) **(SQL, AD)**

- Secure_admin_new_plan as logged event (lists details of plan (name, description) and list boxes displaying selected Service Instances for this plan and ones available within the SLA, button to submit and add service to Plan from SLA list, remove service) **(SQL)**

- Secure_admin_select_services_for_edit (list box for all services, showing which plan and SLA they belong to, buttons to add/modify/delete, Delete as logged event) **(SQL)**

- Secure_admin_edit_service (displays service details (name, description, type), configuration settings (XML), button to submit, as logged event, prompts that this will propagate settings to all affected users) **(SQL, AD)**

- Secure_admin_new_service (see edit service) **(SQL)**

- Secure_admin_select_messages_for_edit (lists all current system messages, buttons for add, delete, move up and move down (in sequence) messages **(SQL)**

- Secure_admin_messages_edit (populates a details panel with message title and message body) **(SQL)**

- Secure_admin_events (lists all events from system log; in the future, this will be searchable)

(SQL)

- Secure_admin_request_review (search, add/edit buttons) **(SQL)**
- Secure_admin_request_edit (if resolved, send an email to the originator) **(SQL)**
- Secure_admin_request_new (request for change of service, defect report, etc.) **(SQL)**

The other pages in the admin site are (with web components in parentheses after):

- `servicetypes.aspx` (all controls in `admin.aspx` PLUS secure_admin_service_types); user must be in EmpowerNet admin group
- `sla.aspx` (all controls in `admin.aspx` PLUS secure_admin_select_SLA_for_edit, secure_admin_edit_SLA, secure_admin_new_plan); users must be in EmpowerNet administrators or support users
- `plans.aspx` (all controls in `admin.aspx` PLUS secure_admin_select_SLA_no_edit, secure_admin_select_plans_for_edit, secure_admin_edit_plan, secure_admin_new_service, secure_admin_new_plan); users must be in group administrators (can select my SLA only) or EmpowerNet administrators or support users (can select all SLAs)
- `services.aspx` (all controls in `admin.aspx` PLUS secure_admin_select_SLA_no_edit, secure_admin_select_plans_no_edit, secure_admin_select_services_for_edit, secure_admin_edit_services, secure_admin_new_service); users must be in EmpowerNet admin or support users group
- `groups.aspx` (all controls in `admin.aspx` PLUS select_admin_select_OU_for_edit, select_admin_edit_OU, select_admin_new_user); user must be in administrators group (can see my OU and below only) or EmpowerNet administrators or support users (support users cannot administer EmpowerNet OU but can see all other OUs)
- `users.aspx` (all controls in `admin.aspx` PLUS secure_admin_select_OU_no_edit, secure_admin_select_user_for_edit, secure_admin_edit_user, secure_admin_new_user); user must be in administrators group (can see my OU and below only) or EmpowerNet administrators or support users (support users cannot administer EmpowerNet OU but can see all other OUs)
- `messages.aspx` (all controls in `admin.aspx` PLUS secure_admin_select_messages_for_edit, secure_admin_messages_edit); user must be in EmpowerNet admin group or support users
- `events.aspx` (all controls in `admin.aspx` PLUS secure_admin_events); user must be in EmpowerNet admin group or support users
- `requests.aspx` (all controls in `admin.aspx` PLUS secure_admin_request_review and secure_admin_request_edit); user must be in an admin group or EmpowerNet support users and can view only the requests they submitted (Empowernet admins and support users can see all requests); can search within allowed requests by the following fields: X, Y, Z (status,

priority). Administrators can edit the following fields: X, Y, Z. ENC admins and support users can edit all fields (including changing status to inactive). Same form for adding a new request.

The normal sequence of operations for provisioning a new customer is:

1. Customer admin sends email to EmpowerNet requesting specific services, identifying initial admin userid preference, and method of transferring data to hosted services (all legal and contractual details are established in a separate, written SLA). The details required from the customer include:

   - Customer account information (customer name, description, bill-to contact name/phone/address/email, admin contact name/phone/address/email)

   - Administrator account information (initial admin account name, long text name, email address, phone number, FAX number, address)

   - Service request (service types, number of concurrent users for each, default plan name and description)

2. EmpowerNet support user is forwarded the request for service and creates the following (in order):

   - Create new Service Level Agreement (`sla.aspx`)

   - Create a new Service Plan under the SLA composed of all instances requested by the customer (`plans.aspx`).

   - Load all customer data, configure application servers, and verify all applications are installed correctly

   - (Add any new Service Types, if necessary; `servicetypes.aspx`)

   - Add new Service Instances under the Service Plan with specific application parameters determined in the previous step (`services.aspx`).

   - Create new group (OU) at root level for the customer, map to the SLA and Service Plan created in the previous steps (`groups.aspx`)

   - Create initial user in the group (OU) as administrator, with the account information specified by the customer request and profile taken from group service plan (`users.aspx`).

   - Notify customer administrator to validate service established.

3. Customer admin receives notification that services are initially provisioned and performs the following optional steps:

   - Create additional Service Plans (grouping existing Service Instances)

- Create additional users
- Create additional sub-groups

7.4 Portal EAI (non-deployed in iteration 1)

TBD.

7.5 Web services (secure)

TBD.

8. Portal Functional Specification (business services)

This section will describe the business logic components.

8.1 Administration operations component

The administrative operations component determines administrative permissions (can user perform operation), executes user requests (add, modify, delete operations), and can provide overall status. The component implements the following functions:

```
Bool addServiceType(strType, strDescription, strXMLSpec)

Bool updateServiceType(strType, strNewType, strDescription, strXMLSpec)

Bool deleteServiceType(strType, blnForce)

XMLString getServiceTypes(void)

XMLString getContentsFromFile(strUNCFilePath)

bool addPrimaryCustomer (strCustName, strCustDesc, strCustBillToName,
strCustBillToPhone, strCustBillToAddress, strCustBillToEmail,
strCustAdminName, strCustAdminPhone, strCustAdminAddress,
strCustAdminEmail)

bool addSecondaryCustomer (gidParentCustomerID, strCustName, strCustDesc,
strCustBillToName, strCustBillToPhone, strCustBillToAddress,
strCustBillToEmail, strCustAdminName, strCustAdminPhone,
strCustAdminAddress, strCustAdminEmail)

bool updateCustomer (gidCustomerID, strCustName, strCustDesc,
strCustBillToName, strCustBillToPhone, strCustBillToAddress,
strCustBillToEmail, strCustAdminName, strCustAdminPhone,
strCustAdminAddress, strCustAdminEmail)

bool deleteCustomer (gidCustomerID, blnForce)

XMLString getCustomer (gidCustomerID)

XMLString getAllSubCustomersOfCustomer (gidCustomerID)

XMLString getAllUsersOfCustomer (gidCustomerID)

bool isCustomerParent (gidCustomerID)

bool deleteAllChildrenOfCustomer (gidCustomerID)  'before deleting a parent
customer, delete it's sub customers

bool deleteAllUsersOfCustomer (gidCustomerID)
bool addService (gidServicePlan, strServiceType, strServiceName,
strServiceDesc, strXMLSpec)

bool updateService (ServiceInstance, strServiceType, strServiceName,
strServiceDesc, strXMLSpec)

bool deleteService (ServiceInstance, blnForce)

XMLString getService (ServiceInstance)

XMLString getAllServicesForCustomer (gidCustomerID) 'gets all services of a
customer's service plan

XMLString getAllServicesForUser (gidUserID) 'all services a user can get

bool addServicePlan (gidServiceLevelInstance, strServicePlanName,
ServicePlanDesc)

bool updateServicePlan (ServicePlan, strServicePlanName, ServicePlanDesc)

bool deleteServicePlan (ServicePlan, blnForce)

XMLString getServicePlan (ServicePlan)

XMLString getServicePlansForMSA (gidServiceLEvelInstance)  'gets all
Service Plans for an MSA

XMLString getServicePlansForCustomer (gidCustomerID) 'gets all ServicePlans
for a Customer (group)

XMLString getServicePlanForUser (gidUserID) 'gets a users ServicePlan

bool addServiceLevelAgreement (gidCustomerID, strServiceLevelName,
strServiceLevelDescription, strPlanRestrictions, blnLoggingEnabled)

bool updateServiceLevelAgreement (ServiceLevelInstance,
strServiceLevelName, strServiceLevelDescription, strPlanRestrictions,
blnLoggingEnabled)

bool deleteServiceLevelAgreegment (ServiceLevelInstance, blnForce)

XMLString getServiceLevelAgreements (void)  'gets a list of all SLA's

XMLString getServiceLevelAgreement (ServiceLevelInstance)  'gets a
particular SLA by ID

XMLString getServiceLevelAgreementByCustomerID (gidCustomerID) 'gets the
customers SLA
```

```
bool addUser (gidCustomerID, gidServicePlan, blnIsAdmin,
strAllowedServices, strProfile, strEmailAddress, strPhoneNumber,
strFAXNumber, strAddress)

bool updateUser (gidUserID, blnIsAdmin, strAllowedServices, strProfile,
strEmailAddress, strPhoneNumber, strFAXNumber, strAddress)

bool deleteUser (gidUserID, blnForce, blnNotify)

XMLString getUser (gidUserID)

XMLString getUserProfile (gidUserID)

bool isUserAdmin (gidUserID)

bool addUserToServicePlan (gidUserID, gidServicePlanID)

bool assignMSAtoCustomer (gidCustomerID, gidServiceLevelInstance)

bool assignServiceToServicePlan (gidServicePlan, gidServiceInstance)

bool removeServiceFromServicePlan (gidServicePlan, gidService)

bool assignServicePlanToMSA (gidServiceLevelInstance, gidServicePlan)

bool removeServicePlanFromMSA (gidServiceLevelInstance, gidServicePlan)

... more to be defined
```

8.2 User business component

The user business component retrieves user information and profile data as well as providing end-user operations for an authenticated user. The component implements the following functions:

`XMLString getUserProfileDOM(uid)` This function calls the Active Directory services component to retrieve the XML DOM that represents the users profile and configuration settings.

`Bool authenticateUser(uid, pwd)` This function calls the Active Directory services component to verify that the user can authenticate against Active Directory.

`Bool isUserAdmin(uid)` This function calls the Active Directory services component to check if the user is marked as an administrator for his/her group (OU).

`Bool isUserSuperAdmin(uid, strENCAdminGroup)` This function compares the user' s group to the one defined in the application configuration as the Empowernet admin group.

`Bool isUserTechSupport(uid, strENCUsersGroup)` This function compares the user' s group to the one defined in the application configuration as the Empowernet users group.

`String getUserGroup(uid)` This function calls the Active Directory services component to retrieve group name (OU) to which the user belongs.

`Bool changePassword(uid, new_pwd, old_pwd)` This function calls the Active Directory services component to change the user password. This may also have to call the SiteMinder or Citrix services components (TBD).

The schema for the user profile file (stored in SQL Server and Active Directory profile attribute) is:

```
<xml>
</xml>
```

9. Portal Functional Specification (data services)

This section will describe the components for accessing different repositories and the application repository schema.

9.1 Active Directory services component

The Active Directory services component authenticates users, and sets and retrieves user profile information in the Active Directory repository. The component implements the following functions:

`XMLString getUserProfileDOM(uid, gid)` Retrieve the XML DOM that represents the users profile and configuration settings.

`Bool authenticateUser(uid, pwd, gid)` Attempt to bind to an AD user entry.

`Bool changePassword(uid, gid, new_pwd, old_pwd)` Change the user password in AD.

`Bool isUserAdmin(uid, gid)` Check if the user is marked as an administrator for the OU.

`String getUserGroup(uid)` Retrieve group id (OU) to which the user belongs.

`String getUserEmail(uid)` Retrieve email address for the user.

`String getUserPhone(uid)` Retrieve phone number for the user.

`String getUserFAX(uid)` Retrieve FAX number for the user.

`String getUserAddress(uid)` Retrieve postal address for the user.

`XMLString getUserSubGroups(uid)` Retrieve sub-OUs which the user might be able to administer.

`String getGroupName(gid)` Retrieve the long text name for the OU.

`String getGroupDescription(gid)` Retrieve the long description for the OU.

9.2 SiteMinder services component

The SiteMinder services component authenticates users, and sets and retrieves user profile information in the SM repository. The component implements the following functions:

`TBD`

9.3 Application data services component

The application data services component manages the SQL Server repository for the application. For specific schema definition, see the Visio diagram. The component implements the following functions:

```
Bool addServiceType(strType, strDescription, strXMLSpec)

Bool updateServiceType(strType, strNewType, strDescription, strXMLSpec)

Bool deleteServiceType(strType, blnForce)

XMLString getServiceTypes(void)
```

`Bool isServiceTypeInUse(strServiceTypeName)` This function checks to see if a service type has any service instances of that type.

bool addService (gidServicePlan, strServiceType, strServiceName, strServiceDesc, strXMLSpec)

bool updateService (ServiceInstance, strServiceType, strServiceName, strServiceDesc, strXMLSpec)

bool deleteService (ServiceInstance, blnForce)

XMLString getService (ServiceInstance)

XMLString getAllServicesForCustomer (gidCustomerID) 'gets all services of a customer's service plan

XMLString getAllServicesForUser (gidUserID) 'all services a user can get

bool addServicePlan (gidServiceLevelInstance, strServicePlanName, ServicePlanDesc)

bool updateServicePlan (ServicePlan, strServicePlanName, ServicePlanDesc)

bool deleteServicePlan (ServicePlan, blnForce)

XMLString getServicePlan (ServicePlan)

XMLString getServicePlansForMSA (gidServiceLEvelInstance) 'gets all Service Plans for an MSA

XMLString getServicePlansForCustomer (gidCustomerID) 'gets all ServicePlans for a Customer (group)

XMLString getServicePlanForUser (gidUserID) 'gets a users ServicePlan

bool addServiceLevelAgreement (gidCustomerID, strServiceLevelName, strServiceLevelDescription, strPlanRestrictions, blnLoggingEnabled)

bool updateServiceLevelAgreement (ServiceLevelInstance, strServiceLevelName, strServiceLevelDescription, strPlanRestrictions, blnLoggingEnabled)

```
bool deleteServiceLevelAgreegment (ServiceLevelInstance, blnForce)

XMLString getServiceLevelAgreements (void)  'gets a list of all SLA's

XMLString getServiceLevelAgreement (ServiceLevelInstance)  'gets a particular SLA by ID

XMLString getServiceLevelAgreementByCustomerID (gidCustomerID) 'gets the customers SLA

bool addCustomer (strCustName, strCustDesc, strCustBillToName, strCustBillToPhone,
        strCustBillToAddress, strCustBillToEmail, strCustAdminName, strCustAdminPhone,
        strCustAdminAddress, strCustAdminEmail)

bool updateCustomer (gidCustomerID, strCustName, strCustDesc, strCustBillToName,
        strCustBillToPhone, strCustBillToAddress, strCustBillToEmail, strCustAdminName,
        strCustAdminPhone, strCustAdminAddress, strCustAdminEmail)

bool deleteCustomer (gidCustomerID, blnForce)

XMLString getCustomer (gidCustomerID)

XMLString getAllSubCustomersOfCustomer (gidCustomerID)

XMLString getAllUsersOfCustomer (gidCustomerID)

bool isCustomerParent (gidCustomerID)

bool deleteAllUsersOfCustomer (gidCustomerID)

bool addUser (gidCustomerID, gidServicePlan, blnIsAdmin, strAllowedServices, strProfile,
        strEmailAddress, strPhoneNumber, strFAXNumber, strAddress)

bool updateUser (gidUserID, blnIsAdmin, strAllowedServices, strProfile, strEmailAddress,
        strPhoneNumber, strFAXNumber, strAddress)

bool deleteUser (gidUserID, blnForce, blnNotify)

XMLString getUser (gidUserID)

XMLString getUserProfile (gidUserID)

bool isUserAdmin (gidUserID)

bool addUserToServicePlan (gidUserID, gidServicePlanID)

bool assignMSAtoCustomer (gidCustomerID, gidServiceLevelInstance)

bool assignServiceToServicePlan (gidServicePlan, gidServiceInstance)

bool removeServiceFromServicePlan (gidServicePlan, gidService)
```

bool assignServicePlanToMSA (gidServiceLevelInstance, gidServicePlan)

bool removeServicePlanFromMSA (gidServiceLevelInstance, gidServicePlan)

`Void logEvent(strEventName, strDescription, strLogonUser)`This function inserts an event in the logged events table (and appends a timestamp for the event).

```
... more to be defined
```

9.4 MSMQ services component

The MSMQ services component sends and receives messages with the MSMQ interface. The component implements the following functions:

```
StatusCode sendMSMQMessage(strServer, strQueue, strMessage)
```

9.5 Citrix services component

The Citrix services component retrieves user subscription information (for display on the application hosting menu) and gets the URL for a particular allowed session. The component implements the following functions:

`XMLString getUserSubscriptions(uid, gid)` Retrieve an XML DOM representing the applications a particular user has subscriptions for.

`String getSubscriptionURL(strSubscriptionInstance)` Retrieve the HTML link string for a given subscription (hosted service instance).

9.6 Application repository schema discussion

The application repository is composed of user, group, and profile information stored in Active Directory combined with a relational SQL Server database for user-group-service plan mappings.