

California Environmental Protection Agency



Air Resources Board

Low Carbon Fuel Standard Program System Development Project System Implementation Plan - Second DRAFT

January 6, 2010

APPROVAL:

Renee Littaua, ARB Project Manager

Signature _____ Date _____

Cindy Stover, FCCC Project Director

Signature _____

Table of Contents

1	INTRODUCTION	1
1.1	Plan Scope.....	1
1.2	Purpose	1
1.3	System Overview	2
1.4	Project References	3
2	IMPLEMENTATION PLAN	4
2.1	Implementation Scope and Strategy	4
2.2	Hardware, Software, and Data.....	5
2.3	Configuration Management.....	9
2.4	System Security	10
2.5	Pre-Implementation Tasks	10
2.6	Production Rollout Tasks	13
2.7	Post Implementation Tasks.....	14
2.8	Production Verification	15
2.9	Contingency/Rollback Plan	16
2.10	System Backup, Restore, and Disaster Recovery	17
2.11	Implementation Team Contact List	18
2.12	System Maintenance	19

-

1 INTRODUCTION

The Low Carbon Fuel Standard (LCFS) System Development project was initiated by the Stationary Source Division (SSD) on January 15, 2009 to fulfill proposed regulatory requirements to provide an information technology system for regulated party reporting on fuels and other data. Regulated parties that are upstream entities (i.e., producers and importers that are legally responsible for the quality of transportation fuels in California) are required to report on fuel carbon intensity values for their fuels, along with the total energy content of each low carbon fuel component replacement for either gasoline or diesel on a quarterly and annual basis. Reporting is to occur via a secure, web-based system through direct data entry and/or an XML upload of data. System functionality includes credit tracking functionality to maintain credits/deficits calculated from submitted fuel data. The system is to securely maintain and report credit/deficit status for each regulated party. Key project milestones include:

- Quarterly progress reporting to begin in 2010 – the first report is due May 31st, 2010 and quarterly thereafter.
- Annual compliance reporting by April 30th starting in 2012 for calendar year 2011

1.1 Plan Scope

This implementation plan covers the implementation of the LCFS system regulated parties facing pages.

The original project requirements include the need for LCFS system ARB facing pages that would support data submission approval, edit, and reporting capability. These ARB facing pages have not been developed and are not included in this implementation plan.

The system development project includes a second phase to develop and deploy credit Bank and Trading System (BTS) functionality to enable regulated parties track the purchases and sale of LCFS credits. The BTS will securely integrate with the Reporting Tool to provide a platform for credit transfers and maintain a credit trading history for each regulated party. This implementation plan does not include the BTS functionality.

1.2 Purpose

Implementation involves preparing the system for production use and typically starts with the creation of an implementation plan. The purpose of the implementation plan is to define and communicate the steps and tasks necessary to install the system into a production environment and prepare the software for production use. The plan defines and describes the tasks and procedures (deployment tasks, implementation activities, mechanisms for resolving outstanding issues, and the strategy to track and monitor implementation status) that must be accomplished to ensure the orderly transition of the system into production operations.

The implementation plan describes the implementation scope and strategy and provides information on the hardware, software, facilities and materials required for the implementation. In addition to defining implementation tasks (broken down into

phases), it identifies contingency plans, system security and the implementation schedule, and typically includes the list of implementation team members. The plan lists the pre-implementation tasks that must be completed before a successful system implementation can be initiated. For each implementation task (e.g., database creation), the detailed steps are listed, the owner (the person or group responsible for completing the task) is named and the estimated start date and time for the activity is identified. Detailed checklists are typically generated to facilitate communication, tracking, and coordination of tasks. The implementation plan also identifies the post-implementation activities that will ensue immediately after cutover as well as contingency plans in the case of a catastrophic event.

Implementation planning can also involve describing maintenance and operations activities, organization, roles, responsibilities, tools and techniques. Maintenance activities involve making corrections to the system to fix bugs as well as implementing enhancements to add new functionality. Maintenance also involves scheduling, testing, and performing software upgrades (e.g., upgrading to a new version of the database software). Operations activities involve the day to day support including help desk, troubleshooting, database backup procedures, and disaster recovery.

1.3 System Overview

This section provides a brief overview of the system to be implemented, including a description of the system and its organization.

1.3.1 System Description

The LCFS system includes regulated parties facing web pages, which provide the external user with the ability to:

- Enter corporation and company profile data
- Establish user accounts
- Enter or import fuel transaction data
- Edit entered fuel transaction data to the point of submission
- Provide revisions to submitted data
- View credit/deficit balance and credit/deficit summary
- Obtain LCFS documentation (compliance and reporting schedules, user manual, etc.)
- Submit feedback and questions to ARB Administrator

1.3.2 System Organization

The LCFS system is a web based application built with Microsoft technologies. Regulated parties access the application using a standard browser (Microsoft Internet Explorer, or Firefox). The user interfaces pages are developed using ASP.NET 3.5

including the use of Ajax to increase the usability of the web pages. Data is stored in a SQL Server 2008 database.

1.4 Project References

This section provides a bibliography of key project references and deliverables that have been produced before this point in the project development.

- Contract - the contract contains the original list of requirements.
- Project Management Plan (PMP) - this plan is being developed at the same time as the Implementation Plan.
- Test Plan - this plan is being developed at the same time as the Implementation Plan.
- Regulatory Party Design Specifications - available in BaseCamp document sharing library. Provides detailed specifications in Visio for the web pages pertaining to Reporting and Credit Balance. Additional design specifications on Organization Profile and Account Management are contained in meeting notes and other documentation available on BaseCamp.
- LCFS Change Request Tracking Tool - Excel spreadsheet which tracks change requests, some of which have been implemented while others are pending.
- LCFS Issues/Risk Tracking – Excel spreadsheet containing a list of identified system and project issues and risks

2 IMPLEMENTATION PLAN

This section provides an overview of the Implementation Plan, including a description of implementation strategy and scope and the activities that must be completed to ensure a successful implementation. Specifically, this section includes:

- ◆ **Section 2.1, Implementation Scope and Strategy**, presents the strategy for system rollout and the organizations that will be involved.
- ◆ **Section 2.2, Hardware, Software, and Data**, lists the items required for successful implementation of the LCFS system.
- ◆ **Section 2.3, Configuration Management**, lists the items required for successful implementation of the LCFS system.
- ◆ **Section 2.4, System Security**, provides an overview of the system security features and system security requirements during the implementation.
- ◆ **Section 2.5, Pre Implementation Tasks**, identifies the components required to be in place before a successful implementation can occur.
- ◆ **Section 2.6, Production Rollout Tasks**, identifies the tasks that will be executed during the cutover period.
- ◆ **Section 2.7, Post Implementation Tasks**, identifies the activities that will ensue immediately after cutover
- ◆ **Section 2.8, Production Verification**, describes the steps involved in verifying the system, once deployed to production, is ready for use.
- ◆ **Section 2.9, Contingency/Rollback Plan**, specifies when the go / no go decision will be made and the factors used to make the decision.
- ◆ **Section 2.10, System Backup, Restore, and Disaster Recovery** describes backup and restore processes.
- ◆ **Section 2.11, Implementation Team Contact List**, provides a list of implementation team members, their role and their contact information.
- ◆ **Section 2.12 System Maintenance**, describes processes and roles involved in Emergency Releases.

2.1 Implementation Scope and Strategy

Implementation of the LCFS regulated party facing pages involves implementing the application on a Acceptance Test server at ARB to allow an industry focus to perform acceptance testing in January 2010 as well as implementing the application on two servers (database server and web server) at Raging Wire's data center for production use starting February 1st 2010. To achieve this aggressive schedule many implementation tasks are being performed in parallel. Daily "stand-up meetings" are

being held, via a conference call, to focus on action items and to keep the many concurrent tasks moving.

2.2 Hardware, Software, and Data

There are four environments in use:

1. **Development environment** used by FCCC developers and testers.
2. **Test environment** used by ARB staff to perform system testing.
3. **User Acceptance Test (UAT)** used by the industry focus group to perform acceptance testing.
4. **Production environment.**

Exhibit 1 shows the current development environment which includes a single server (FCCC is performing development work for other projects on this server.) This environment currently exists.

Exhibit 1 - Development Environment

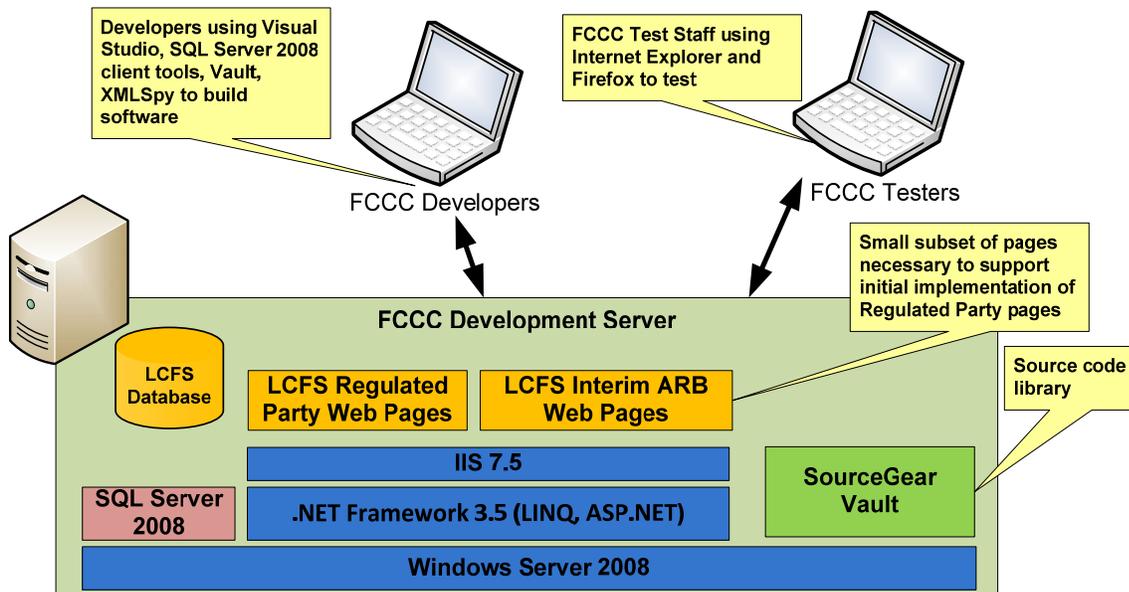


Exhibit 2 shows the current development environment which includes a single server (FCCC is performing development work for other projects on this server.) This environment currently exists.

Exhibit 2 - System/Integration Test Environment

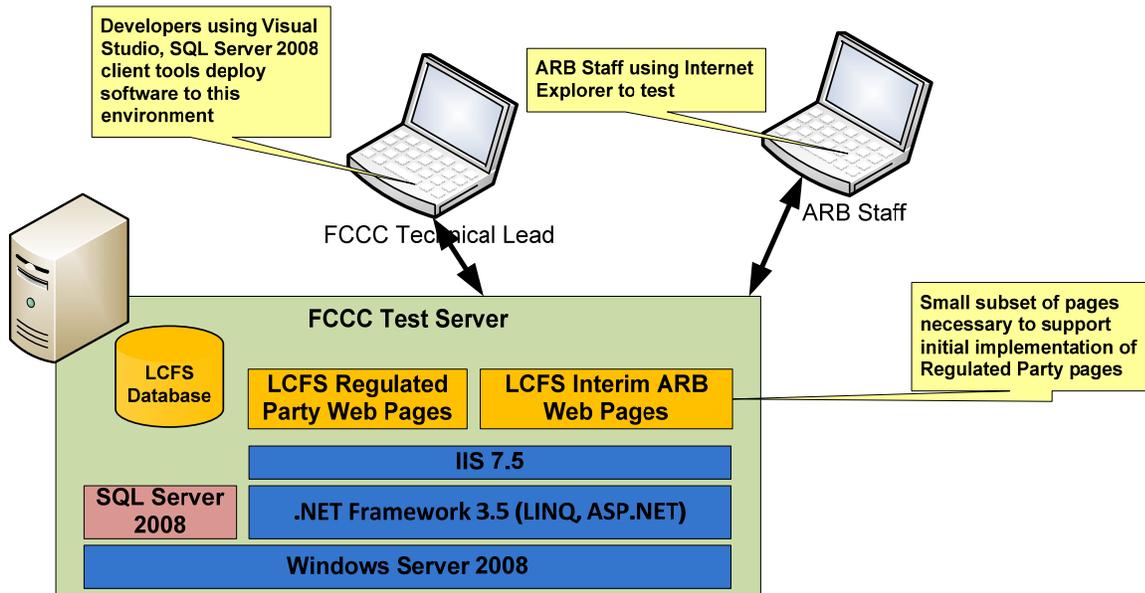


Exhibit 3 shows User Acceptance Test (UAT) environment which includes a single server in ARB's server room.

Exhibit 3 - User Acceptance Test (UAT) Environment

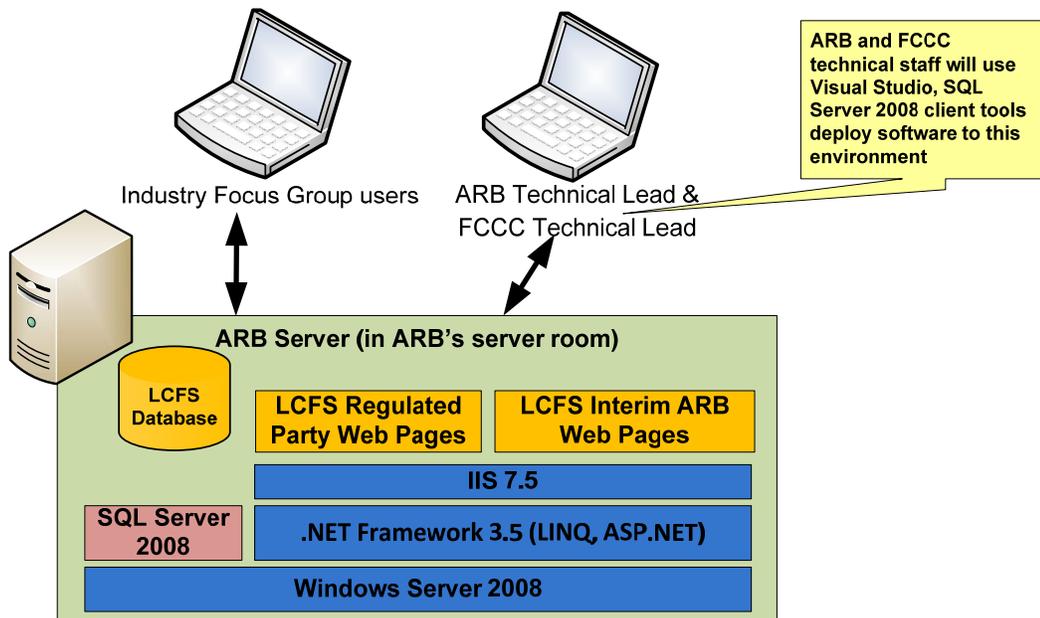
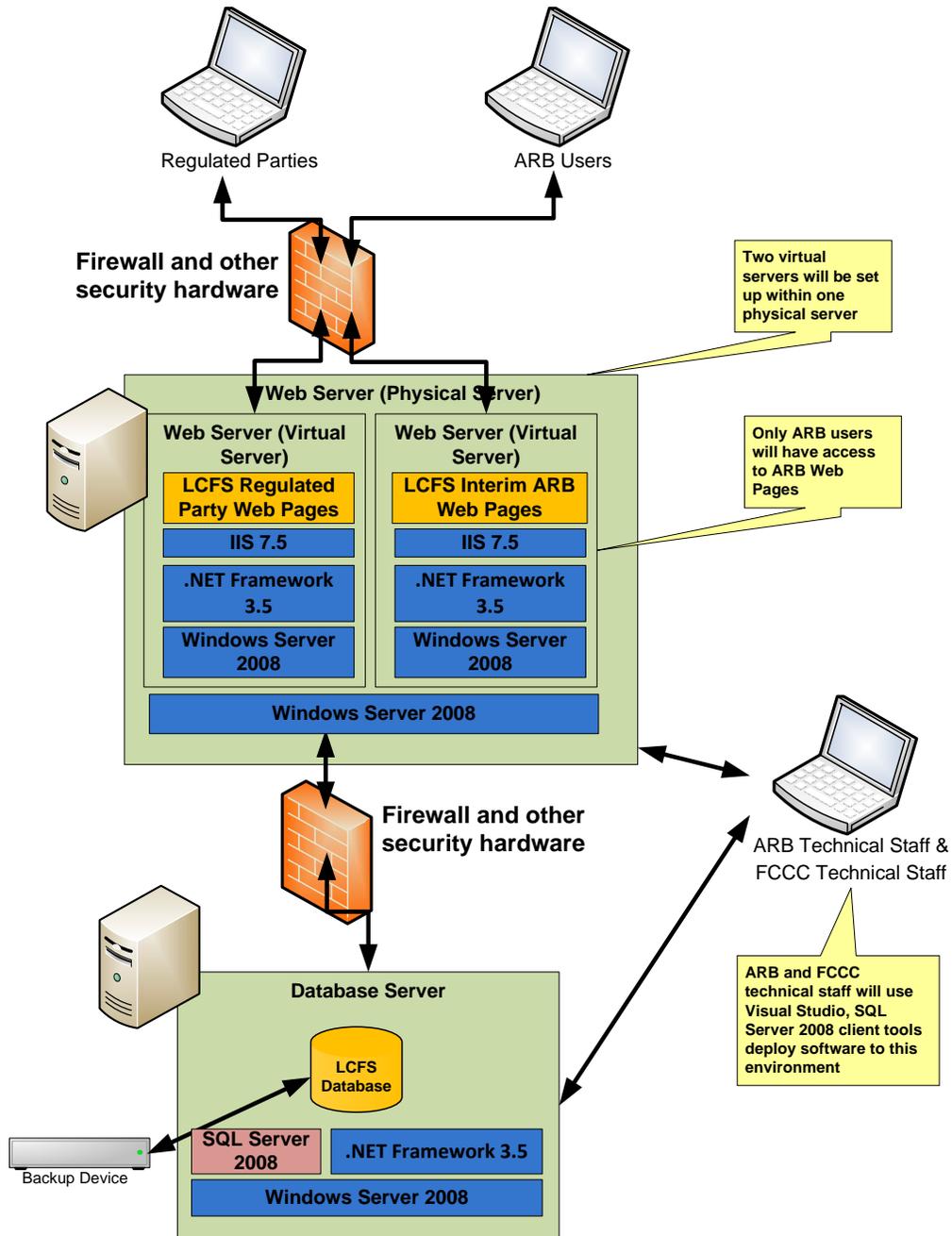


Exhibit 4 shows a potential Production environment which includes separate web and database servers. Note that two virtual web servers will be used to ensure separation of the ARB facing pages; only ARB users will have security privileges to access the server containing the ARB pages. This environment will be hosted at Raging Wire's data center.

Exhibit 4 - Production Environment



2.2.1 Hardware

This section describes the hardware requirements necessary to support the implementation.

- ◆ **Development server**, which is located at FCCC and is used to support the development environment shown in Exhibit 1 as well as the Test Environment shown in Exhibit 2.
- ◆ **User Acceptance Test Server**, which is located in ARB's server room.
- ◆ **Production Web Server**, located in Raging Wire's data center.
- ◆ **Production Database Server**, located in Raging Wire's data center.
- ◆ **Backup hardware** that is provided by Raging Wire at their data center as part of the hosting arrangement to support the Production Web Server and Production Database Server.

2.2.2 Software

This section describes all software required to implement the system including commercial software and custom software.

Third party software components include:

- ◆ **Windows Server 2008** w/IIS enabled (Application Server role enabled, ASP.NET enabled). The operating system that is installed on all servers (web servers and database servers).
- ◆ **SQL Server 2008 SP1**. The relational database management software that is installed in all environments. In production this software is installed on the database server.
- ◆ **.NET Framework 3.5** (w/SP1 is ok, but not necessary). The Microsoft application framework that is installed in all environments on all servers. . In production this framework is installed on the web server and database server.
- ◆ **AjaxControlToolkit.dll** v.3.0.30930.0. An open source dynamic link library used to provide user interface functionality that is installed in all environments. . In production this library is installed on the web server.
- ◆ **CuteWebUI.AjaxUploader.dll** v.3.0 20090518 (licensed for up to 10 domains). An dynamic link library that allows users to upload files to the server. This library is installed in all environments. In production this library is installed on the web server.
- ◆ **Microsoft Explorer 7or 8, Firefox 3.5 or greater**. Users and support staff must use one of these browsers to access the application.
- ◆ **Altova XML Spy Professional**. This is used on user workstations to validate XML files before they are imported into the application.

- ◆ **SourceGear Vault** - This is the source code control software that is used by FCCC's developers to store the LCFS .NET source code. This product is only installed in the development environment.

Custom software includes:

- ◆ **Regulated Party facing pages** built in C# for ASP.NET
- ◆ **Interim ARB facing pages** built in C# for ASP.NET (these are the minimal subset of pages needed to support the initial implementation of the Regulated Party Pages)
- ◆ **XSD**
- ◆ **SQL Server database** including tables, stored procedures, primary keys, foreign keys.
- ◆ **.NET configuration files**

2.2.3 Data

This section describes specific data preparation requirements and data that must be available for the system implementation.

LCFS reference data (e.g. fuel pathways) is created in Excel spreadsheets within BaseCamp and loaded into the SQL Server database. System-generated messages and emails are created as reference data in the database.

Reference data will also include a list of biofuel companies that are registered through an external process using an Excel spreadsheet. Data from registered biofuel companies will need to be uploaded into the database prior to production.

2.3 Configuration Management

This section describes the interactions required with the Configuration Management (CM) representative on CM-related issues, such as when software listings will be distributed, and how to confirm that libraries have been moved from the development to the production environment.

FCCC is storing all .NET source code within SourceGear Vault on the development server.

The SourceGear Vault labeling feature is used to identify source code within a particular release of the application.

.NET code will be published from Visual Studio

A database backup, loaded with just reference data, will be used to create the initial UAT and Production databases

Step-by-step deployment instructions are included in the appendix of this document.

2.4 System Security

2.4.1 System Security Features

Each regulated party will be assigned a unique user id and password for main user. This is then enabled by ARB (a database field). Once enabled this user can create other user accounts for the same regulated party.

Role based security is being used. The roles are:

- Reviewer – have only read access to the system; may or may not have signatory authority
- Contributor – have read and write access to the system; may or may not have signatory authority
- Administrator – have read, write, and account management authority; may designate other administrators or users with signatory authority;
- Signatory – users have the ability to submit report on behave of the company; this user may be a Reviewer, Contributor, or Administrator and must accept this designation manually

Data for all regulated parties is stored in a single SQL server database. Application logic controls access to ensure a regulated party can only view and update their own data.

XML upload loads data into a user interface web page which goes through the same security mechanisms as data entered directly on the web page.

2.4.2 Security during Implementation

The LCFS will be preloaded with reference data (e.g. fuel types, fuel pathways) but will not contain regulated party data until the users access the system and enter data. The majority of the data that will be preloaded into the system is not considered to be sensitive. However, bio-fuel company data, including profile, carbon intensity, and physical pathway need to be preloaded. This bio-fuel data is considered sensitive. Access to the production SQL Server database will be controlled by only allowing Database Administrators to directly access the database, therefore the security risks during actual implementation are minimal.

2.5 Pre-Implementation Tasks

The purpose of the pre-implementation task is to achieve production readiness.

Exhibit 4 lists the pre-implementation tasks that must be completed before a successful LCFS system implementation can be initiated. For each task, the detailed steps are listed, the owner (the person or group responsible for completing the task) is named and the estimated start date and time for the activity is identified. Many of these tasks will occur concurrently.

ARB is the final Reviewer/Approver on all Pre-Implementation and Production Rollout Tasks and in general ARB staff will be involved in all technical aspects of this project (except code details) as ARB staff will be supporting the system long term including both operation and maintenance.

Exhibit 4 - Pre-Implementation Tasks.

Task	Owner	Reviewer/ Approver	Date/ Time
Identify UAT environment 1. Identify UAT Server	ARB	ARB	12/31/2009
Identify Training environment 1. Will training be performed in production environment? 2. If separate environment add new implementation tasks.	ARB	ARB	12/31/2009
Acquire Production Servers and other Hardware 1. Identify data center (Raging Wire) 2. Specify web server, data server, and backup requirements 3. Write internal ARB FSR 4. Gain approval from ARB IT and executives 5. Prepare and execute contract with Raging Wire	ARB	ARB	1/15/2010
Acquire Software 1. Identify licensing needs (SQL Server, Windows Server) 2. Identify Domain registration and DNS needs.	ARB	ARB	1/15/2010
UAT Server setup and application deployment (Assuming Windows Server 2008 preinstalled) 1. Install and configure SQL Server 2008 2. .NET framework installation/configuration 3. Install third-party DLLs 4. IIS configuration 5. Account (admin and service accounts) setup 6.	ARB and FCCC Technical Leads	ARB	1/22/2010
Production Database Server initial setup (Assuming Windows Server 2008 preinstalled) 1. Install and configure SQL Server 2008	ARB DBA	ARB	1/22/2010

Task	Owner	Reviewer/ Approver	Date/ Time
Production Web Server initial setup (Assuming Windows Server 2008 preinstalled) 1. Set up Virtual Server for Regulated Party Facing Pages <ol style="list-style-type: none"> a. .NET framework installation/configuration b. IIS configuration c. Account (admin and service accounts) setup d. DNS setup 2. Set up Virtual Server for ARB Facing Pages <ol style="list-style-type: none"> a. .NET framework installation/configuration b. IIS configuration c. Account (admin and service accounts) setup d. DNS setup 	ARB Server Administrator	ARB	1/22/2010
Distribute XML/XSD 1. Make available to industry	ARB Project Lead	ARB	12/31/2009
Identify and document help desk procedures 1. Identify primary point of contact 2. Identify technical support from FCCC, data center, .etc 3. Escalation process 4. Communication process if application is down for an extended period	ARB Project Lead	ARB	1/22/2010
Identify and document system backup 1. Define system back-up and maintenance requirements for all environments <ol style="list-style-type: none"> a. Define back-up schedule b. Define back-up scope (data, ad hoc query tools, etc) 2. Assign responsibilities 3. Implement system back-up and maintenance activities	ARB Database Administrator, FCCC Database Administrator	ARB	1/15/2010
Define & Implement Configuration Management for each Environment 1. For .NET components, define deployment process	ARB and FCCC Server Administrators, Database Administrators	ARB	1/15/2010

Task	Owner	Reviewer/ Approver	Date/ Time
2. For database, define process 3. Implement processes			
Define Post Implementation Verification Test Requirements 1. Define activities or tests to determine if the implementation was successful 2. Identify user id that will be used in production for verification testing. 3. Define the process for creating an “action item” list for any open issues	ARB Project Lead	ARB	1/8/2010
Complete UAT Testing	ARB & Industry Focus Group	ARB	1/29/2010
Demonstration of System to Regulated Parties 1. Plan for demo. Script for demo. 2. Identify Environment to be used for demo 3. Load version of system/database 4. Preload data? 5. Dry run of demo	ARB Project Lead, FCCC Project Manager, FCCC	ARB	1/21/2010

2.6 Production Rollout Tasks

Once all the pre-implementation tasks have been successfully completed the system can be implemented in production. Exhibit 5 lists the production rollout tasks.

Exhibit 5 - Production Rollout Tasks

Task	Owner	Reviewer/ Approver	Date/ Time
Create/Deploy production database 1. Run script to build the database (assumes scripts are being used to create database) 2. Run script to load reference data into database	ARB DBA	ARB	1/27/2010
Deploy .NET components to production Web server 1. Deploy .NET application components for Regulated Party Facing pages to Virtual Server 1 2. Deploy .NET application components for ARB	ARB Server Administrator	ARB	1/27/2010

Task	Owner	Reviewer/ Approver	Date/ Time
Facing pages to Virtual Server 2			
Perform System Back-Up 1. Implement system back-up	ARB DBA	ARB	1/27/2010
Perform Post Implementation Verification 1. Conduct activities or tests listed in Section 2.8 to determine if the implementation was successful	ARB Project Lead ARB DBA	ARB	1/28/2010
Make go/no-go decision 1. To make the “go” decision the production verification steps must be successful. Only previously agreed upon defects (e.g. minor severity) can exist within the system. 2. If “no-go” see section 2.9 for contingency/rollback steps	Team	ARB	1/28/2010
Make system available to end users 1. If “go” decision then make system available to users. <<Need to define exactly how this will be done >>	ARB Server Administrator	ARB	2/1/2010

2.7 Post Implementation Tasks

Once the system is in production it is very important to proactively manage the production system. This includes quick and appropriate response to support requests to avoid small issues becoming larger issues. Performance will be closely monitored initially. System logs will be proactively reviewed to look for problems. Support requests will be analyzed to identify patterns etc. A post implementation review will occur and be documented to provide lessons learned for continuing improvement and as input to the next project.

Exhibit 6 - Post-Implementation Tasks

Task	Owner	Reviewer/ Approver	Date/ Time
Provide Training to Industry 1) Plan for training 2) Create training materials 3) Give training session(s)	FCCC and ARB	ARB	After 2/1/2010

Task	Owner	Reviewer/ Approver	Date/ Time
Evaluate User Support Requests 1. Evaluate user support requests 2. Determine if additional training / documentation or system modifications are required 3. Identify and track tasks and assign responsibility 4. Follow-up to ensure resolution	ARB PM FCCC PM	ARB	Once a week in Tuesday status meeting starting 2/9/2010
Evaluate System Performance 1. Monitor system performance using various tools 2. Conduct tuning to improve performance 3. Check system logs on Database server and Web server for errors	FCCC DBA ARB DBA	ARB	Continuously for first week starting Monday 2/1/2010
Perform System Back-Up 1. Automated backup	None (automated)	ARB	None (automated)
Verify System Back-Up 1. Verify backups are working by restoring database to another location	ARB DBA	ARB	2/3/2010
Conduct Post Implementation Review 1. Review functionality 2. Review availability 3. Review performance	Team	ARB	3/1/2010

2.8 Production Verification

This section describes the process for reviewing the implementation and deciding if it was successful. It describes how an action item list will be created to rectify any noted discrepancies. These steps are performed as part of implementation before the system is released to the public. The purpose of this verification is not to perform a complete system test but to verify that the various components have been moved to the production environment. This process occurs prior to the go/no-go decision.

Functionality verification steps which will be performed by an ARB user:

1. Verify login as different types of users
2. Enter/view organization/user details

3. Create a report (using a special user id)
4. Create data and verify calculations
5. Submit a report (using Signatory user id)
6. Verify XML upload (using a special user id)
7. Verify interim ARB facing pages-
8. Verify credit balance and summaries

Database verification steps which will be performed by a database administrator:

1. DBA verifies that reference data is loaded.

2.9 Contingency/Rollback Plan

This section specifies factors to be included in making the go/no-go decision. For the first production release, in February 2010, there is no need for a rollback plan because the LCFS regulated party facing pages is a new system with no prior system. Therefore, if there is a no-go decision there is no original system to restore. However, once a version of the application has been released and there is a new version to release into production and rollback plan is needed to allow for the situation where the new build is deployed to production but fails the production verification steps (i.e. the new version of the application does not work correctly). Exhibit 8 contains the tasks that need to be performed to restore the previous version of the application.

Exhibit 8. Rollback Tasks

Task	Owner	Reviewer/ Approver
Deploy .NET components to production Web server 1. Remove .NET application components from web server 2. Deploy .NET application component for previous build to web server	ARB Server Administrator	ARB
Restore Database Backup 1. If the application deployment involved updates to the database then restore the previous system backup.	ARB Database Administrator	ARB
Perform Post Implementation Verification 1. Conduct activities or tests listed in Section 2.8 to determine if the implementation was successful	ARB Project Lead ARB DBA	ARB
Make system available to end users	ARB Server Administrator	ARB

2.10 System Backup, Restore, and Disaster Recovery

The following section describes the system back-up processes, including data storage, data back-up management and organization, and data recovery.

2.10.1 System Backups

The following system backups will be performed

- ◆ A backup of all source code and supporting files (including third party DLLs and configuration files) will be performed for each release of system
- ◆ Daily production database backups will be performed using SQL Servers backup utility.
- ◆ Backup tapes will be taken offsite periodically
- ◆ <<More details to be added to this section once Raging Wire production environment hosting details are known>>

2.10.2 Disaster Recovery

This section describes the back-up and recovery processes for disaster recovery (a major disaster requiring restoration of the entire database).

The LCFS system regulated party pages are used for quarterly reporting. Users are not expected to be accessing the system on a daily basis. The exception to this is during the few days prior to a reporting deadline when many (perhaps the majority) of users will be using the system to enter their reports. During this time period having the system down for any extended period (e.g. more than an hour) would cause considerable issues.

Offsite backups of the database and all of the software components used to implement the system will be maintained. Disaster recovery will involve procuring, and configuring new servers, installing the most recent version of the application, and restoring the most recent backup. <<More details to be added to this section once Raging Wire production environment hosting details are known>>

2.10.3 Recovery of Corrupt or Deleted Data

This section describes the back-up and recovery processes for corrupt or deleted data.

Data corruption and deletion can occur for a number of reasons, such as software bugs, software viruses, mistakes made by database administrators. The LCFS application does not allow users to delete data and database administrators should not delete data, but they have to security permissions that allow them to delete data.

The LCFS system stores data in a SQL Server 2008 database. SQL Server backup utility will be used to perform regular backups of the LCFS data.

In the undesirable situation where data is corrupted or deleted recovering data will involve using the database backups to restore data. The ARB staff supporting the LCFS application will need to meet and discuss the best approach for data recovery.

The first step will be to make the LCFS application unavailable to users so that the database is not being updated.

The next step should be to perform a full database backup. This backup is important to have in case mistakes are made while staff are trying to recover data.

Various options are available for recovering data, and the team will need to decide which is the best after reviewing the situation.

A full database restore will restore data for all regulated parties. If the entire database has been corrupted then this is appropriate action.

If partial database corruption has occurred, other, more manually intensive options are available. The database administrator can restore parts of the database.

A full database restore to a temporary database could be performed and then selective records retrieve and updated in the primary database but this would require manual database administrator steps.

2.11 Implementation Team Contact List

Exhibit 9 lists the people who are involved in implementation. Note that a person's role listed in this exhibit is their role within the implementation effort, which may be different to their role on the overall project.

Exhibit 9. Implementation Roles and Responsibilities Matrix

Role	Name	Phone	Email
FCCC Project Manager	Cindy Stover	(916) 325-1850	cstover@foundationccc.org
ARB Project Manager	Renee Littaua		rlittaua@arb.ca.gov
FCCC Configuration Management Manager	Blair Varley	916-276-1190	blair_varley@yahoo.com
FCCC Database Administrator	Blair Varley	916-276-1190	blair_varley@yahoo.com
FCCC Server Administrator	Blair Varley	916-276-1190	blair_varley@yahoo.com
ARB Project Lead - testing - production verification	Christina Zhang-Tillman	916-324-0340	czhangti@arb.ca.gov
ARB Project Lead	Greg O'Brien	916-323-0023	gobrien@arb.ca.gov

Role	Name	Phone	Email
- contract - environment procurement/contract			
ARB Database Administrator	Greg O'Brien	916-323-0023	gobrien@arb.ca.gov
ARB Server Administrator	Nate Black	?	?
FCCC PM Advisor	John Gray	(916) 833-1618	jgray@infiniti-consulting.com
FCCC PM Advisor	Carolyn Borden	916-741-9517	cborden@arb.ca.gov
FCCC Test Manager	Doug Buchanan	?	dbuchanan@capacd.com

2.12 System Maintenance

Immediately after the system is successfully implemented in Production the System Maintenance phase begins. It is important to define the people and processes that will be followed for controlling and implementing Emergency Releases, if they are necessary, as well for new release containing additional functionality.

2.12.1 Emergency releases

Emergency release should follow an abbreviated form of the implementation plan including pre-implementation, implementation, and post-implementation tasks as listed in Exhibit 10 below.

Exhibit 10. Emergency Release Tasks

Task	Owner	Reviewer/ Approver
Identify and document cause of issue 1. ARB documents and analyses issue	ARB	ARB
Meet to discuss resolution 1. ARB and FCCC discuss how to resolve issue 2. FCCC estimates work effort involved in coding and testing	ARB PM, FCCC PM, ARB Project Lead, FCCC technical staff	ARB
Make code changes and test 1. Make code changes 2. Make database changes (scripts), if necessary	FCCC Developers	ARB

Task	Owner	Reviewer/ Approver
3. Perform unit testing and integration testing 4. Document changes		
Perform System Testing 1. Update test scripts 2. Deploy the new build to the test environment 2. Test the specific defect correction 3. Perform regression testing of the application to help ensure nothing else has been broken as part of the fix- if under time crunch team will need to agree on extent of regression testing.	ARB and FCCC	ARB
Communicate down time 1. Communicate down-time to users (if necessary). It is expected that there would be a three hour window when the system is unavailable to allow a new release of the application to be deployed and verified before making it available. This downtime would be coordinated to minimize impact to users.	ARB	ARB
Deploy new build 1. Make system unavailable to users 2. Deploy new build to production environment	ARB Server Admin, ARB DBA	ARB
Perform Production Verification 1. Verify new build using production verification steps list in section 2.8	ARB Project Lead	ARB
Make go/no-go decision 1. Make go/no go decision	ARB and FCCC	ARB
If “no-go”, execute rollback plan If “no-go”, then deploy previous build following steps listed in section 2.9	ARB Server Admin, ARB DBA	ARB
Make system available to end users	ARB Server Administrator	ARB
Communicate “system is up” Communicate to stakeholders (e.g. help desk) and, if appropriate, to users letting them know system is available.	ARB Project Lead	ARB

2.12.2 Change Requests

Change Requests will be processed as described in Section F (Scope Management) of the Project Management Plan. When change requests are made to the application a new version of the application will be produced; the tasks involved in releasing this new version of the application are the same as those listed in previous section for an emergency release. However, a major difference is that more time is available for system testing (regression testing) and the date/time the new version is deployed can be scheduled and communicated weeks or months ahead of time.

Appendix A - Web Site Deployment steps including database backup and restore.

The following steps document how FCCC's developers currently deploy the application to development and test servers. These steps provide the basis for how the application will be deployed to UAT and Production. However, because UAT and Production environments are different (for example, production contains separate web server and database server) the deployment steps will be different. Consequently this list will be refined and documented in more detail once the details of how the software will be deployed to UAT and production are finalized.

1. Publishing the Web Site

- 1) Publish LCFS Visual Studio C# Project for every new build. After publishing the project, all compiled code is stored in "PreCompiledWeb" folder which is located under the project folder. For instance, if the project folder is "C:\LowCarbonWeb" then, the precompiled folder will be located at "C:\LowCarbonWeb\PreCompiledWeb".
- 2) To deploy the new build on the server, notify everyone who might be using the web site that it will be down for X minutes or X hours due to the upgrade. Or deploy the new build after office hours when the team members will less likely to be using the system.
- 3) Remote log into the web server where the project will be deployed.
- 4) Browse to the destination folder on the server's local drive where the web project is located. For instance, C:\inetpub\TestLowCarbonFuel
- 5) Browse to the source folder on the development workstation where the web project is located. For instance, C:\LowCarbonWeb\PreCompiledWeb.
- 6) If this is not the first time deployment, backup the "web.config" file at the destination folder. This is necessary if the database name on the development workstation is different from the one on the server. Otherwise, it is optional.
- 7) If this is not the first time deployment, backup the "bin" folder at the destination folder. As "Ajaxuploader.lic" file does not get included in the bin folder while publishing the web site on the source computer.
- 8) Copy all the files and folders in the source folder into the destination folder. Overwrite all files if this is not the very first deployment.
- 9) If this is the very first deployment, open the "web.config" file in the destination folder to make sure that it has the correct database name which matches the actual database name which will be deployed. Also, copy back the "Ajaxuploader.lic" from the backup bin folder to the bin folder.
- 10) If this is the very first deployment, the SQL Server 2008 database needs to be restored on the database server. Please refer to Section 2. Otherwise, this step can be ignored. If the database schema changes involved with the new deployment, please refer to Section 3.

2. Backup and Restore the Database for Initial Deployment

- 1) Backup the database on the source computer. For instance, it could be on the development box.
- 2) Remote log into the database server.
- 3) Browse to the source folder where the database backup file is located
- 4) Browse to the destination folder where the database backup file is to be placed.
- 5) Go the Microsoft SQL Server 2008 Management Studio. Select the database server where a new database will be created. Expand the plus sign next to it. Browse to "Databases" folder and right click and choose "New Database". Type in the new database name desired in the dialog provided. For example, "LowCarbonFuel".
- 6) Right click on the newly created database and choose Tasks -> Restore -> Database from the context menu. A dialog will appear and select "From Device" radio option on there. Browse to the folder where the database backup file is located (refer to Step 4). Click OK to complete restoring of the database.

3. Making Database Schema Changes after Initial Deployment

Once the database is up and running, to make changes for the database schema will involve the running SQL scripts first ran in the Test environment and then going through a planned testing process. Production Database down time could be required depending on change.