# Table of Contents

- FSFN Application Architecture........................................................................................................... 3
- Technologies Involved............................................................................................................................ 4
  - Core Technologies – J2EE and the Web Browser.................................................................................. 4
  - Internet Explorer Browser...................................................................................................................... 4
  - Jakarta Struts ........................................................................................................................................ 6
  - Model View Controller and Struts.......................................................................................................... 6
  - FSFN Framework Services ..................................................................................................................... 7
  - Batch Architecture ............................................................................................................................... 8
  - FSFN Security Architecture.................................................................................................................. 8
- FSFN Authentication through LDAP ....................................................................................................... 9
  - Tivoli Identity Server ............................................................................................................................ 10
  - FSFN Authorization............................................................................................................................... 11
- FSFN Release 1 Environments, High Level Overview ............................................................................. 13
  - On-line Application Environment....................................................................................................... 13
  - Batch and Reporting Environment...................................................................................................... 14
  - Interface Environment ......................................................................................................................... 17
- FSFN Release 2A Environments, High Level Overview ........................................................................... 19
  - On-line Application Environment....................................................................................................... 19
  - Batch and Reporting Environment ...................................................................................................... 22
  - Interface Environment ......................................................................................................................... 23
- FSFN Release 2B Environments, High Level Overview ........................................................................... 24
  - On-line Application Environment....................................................................................................... 25
  - Batch and Reporting Environment ...................................................................................................... 28
  - Interface Environment ......................................................................................................................... 29
- Remote Data Capture (RDC) Device Server Side Architecture ............................................................... 33
  - Security .................................................................................................................................................. 34
  - Web Service Implementation .................................................................................................................. 35
  - Web Service Message Implementation .................................................................................................. 36
FSFN Application Architecture

The FSFN application implements the Model-View-Controller (MVC) design pattern using Java 2 Platform Enterprise Edition. MVC is an industry standard design pattern used to construct applications with a graphical user interface. The framework facilitates the implementation of the chosen n-tier architecture as shown below.

The FSFN Application Framework is non-proprietary, and is based upon the Apache Jakarta Struts Framework (Struts). FSFN has added on to the component architecture that is provided by Jakarta Struts, supplying fundamental Framework Services that are critical to the operation of enterprise systems. The following sections describe in more detail Jakarta Struts and the FSFN Framework Services.

To understand the framework support for the “n” tier architecture, it is important to recognize the various technologies involved in the implementation. The presentation layer runs on the client workstation and is typically a browser (Microsoft Internet Explorer) but in certain scenarios where word processing is needed (letters and notices), the client is Microsoft Word. All remaining layers, used for business rules enforcement, data validation, and data access and process control are implemented on the server. The process control is also partly implemented in the user interface by means of a navigational design. The overall picture of various underlying technologies is as shown in the diagram below.
Technologies Involved

Core Technologies – J2EE and the Web Browser

FSFN uses the J2EE specification for the development standard for the server side layers. This allows for platform independence high scalability, high security, multi-vendor support and most important - open standards. J2EE is a collection of various technologies and following is the list used in the project:

- JDK 1.4.2 – Java development toolkit for all server side layers;
- JDBC 2.0 – Java database connectivity for data access;
- JSP 1.1 – Java server pages for server side rendering of client pages; and
- Servlet 2.3 – Server side controller specifications

Due to the rich nature of the user interface, various client technologies are used. The implementation of the client interface is for a browser however this has a collection of various technologies:

- RTF 1.6 and ActiveX using Visual Studio 6.0 – For rendering Word documents to the user
- JavaScript 1.2 and HTML 4.0.1 – For the rich browser user interface

Internet Explorer Browser

Microsoft Internet Explorer is the chosen browser for FSFN. There are other browser specific features exploited as a part of the FSFN application design. This is the use of modal windows and modeless windows. The modal and modeless windows channel the FSFN user navigation on the client side to provide process control and step-by-step data entry process. iframes are used for back channel communication to save network overhead and ease of development.

JavaScript is used as glue to tie the user interface components together. The use of JavaScript is minimal except for setting values or obtaining data from modal windows. This improves the performance on the client side and eliminates timing dependencies on the client workstation.

ActiveX technology is also used in conjunction to HTML and JavaScript primarily for interaction with Microsoft Word. Active X is used to Internet enable the Word based office automation component of the FSFN application, providing features such as spell check and automated save of documents to the centralized FSFN document repository. The use of this technology allows transfer of data to the client by means of HTTP(s) and not putting any additional firewall holes or burden on the infrastructure. The only
The requirement is a licensed version of Microsoft Word 97 or higher on the client. The Internet standard distribution mechanism is used to install the ActiveX control on the client workstation. The code is digitally signed and packaged along with the application archive and no additional network support is required.
Internet Explorer through the DCF VPN/Citrix

FSFN is a web application hosted within DCF's intranet. Outside access is controlled by connecting to the DCF VPN and accessing the application through Citrix. The application is fully functional through Citrix as long as the proper release of Internet Explorer is installed and valid licenses for Microsoft Word are available on the Citrix machine.

Jakarta Struts

The Jakarta Struts Framework is the core for the entire FSFN Online Application framework. Struts is an open source Java-based framework developed by Jakarta, which is a branch of the Apache foundation software sponsored by IBM. The framework is built on collection of rich, lean and industry standard technologies to improve developer productivity. The framework also enforces strong pattern based development for consistency across the development environment.

Struts has been written to support any web application using the MVC pattern. However due to the unique needs of the FSFN application, additional core functionality is needed on top of what Struts provides.

Model View Controller and Struts

The FSFN MVC architectures separate core data access functionality from the presentation and control logic that uses this functionality. Such separation allows multiple views to share the same enterprise data model, which makes supporting multiple clients easier to implement, test and maintain. It also supports robust security measures, promotes reuse of code for multiple functions, and enables tuning for optimal performance.

The model represents enterprise data and the business rules that govern access to and updates of this data. Often the model serves as a software approximation to a real-world process, so simple real-world modeling techniques apply when defining the model.

A view renders the contents of a model. It accesses enterprise data through the model and specifies how that data should be presented. It is the view’s responsibility to maintain consistency in its presentation when the model changes. This can be achieved by using a “push” model, where the view registers itself with the model for change notifications, or a “pull” model, where the view is responsible for calling the model when it needs to retrieve the most current data.

A controller translates the interactions with the view into actions to be performed by the model. In the stand-alone GUI client, user interactions could be button clicks or menu selections, whereas in a Web application, they appear as GET and POST HTTP requests. The actions performed by the model include activating business processes or changing the state of the model. Based on the user interactions and the outcome of the model actions, the controller responds by selecting an appropriate view.
The diagram below displays the FSFN implementation of the MVC architecture.

**Exhibit 4.2-1: Model View Controller in FSFN**

Every functional component of the FSFN system follows the above architectural pattern. The Framework provides the components that are surrounded by a solid line in the above diagram itself. The components that are surrounded by dotted lines in the above diagram need to be developed by an application developer. These items include:

- A JSP, that provides the user view of the functionality;
- An Action Class, that provides the controller functionality for a functional set of transactions;
- An Action Form Class, that acts as the data container for a functional set of transactions. The Action Form Class also provides User Interface Validation Logic;
- A Business Logic Class, that controls and manages the complex business rules for a functional set of transactions; and
- A Data Access Class, that provides database access and a persistence mechanism for a functional set of transactions.

**FSFN Framework Services**

In addition to Struts, there are common services provided by the FSFN application framework. The framework uses a Façade pattern to hide the implementation of its services and to give users of the framework a one face “Foundation”. All requests for primary Framework services are channeled through the Façade. The Façade delegates the service request to the appropriate service provider. By hiding the actual implementation, it’s easy for the framework to enhance features or totally replace logic behind the scene. This is extremely beneficial during the maintenance phase.

The key framework services include:

- Data Cache and relevant interfaces;
- Security provider interface;
- Exception and error handling, error substitution;
• Dynamic outliner;
• Database connection and transaction management related API’s;
• Encryption/Decryption API’s;
• Base 64 encoding/decoding API’s;
• Application server API;
• State management;
• Transport management;
• Email;
• Transaction statistics; and
• Office automation.

**Batch Architecture**

The FSFN batch architecture provides the functionality to execute offline processing. Although this processing is not as apparent to the end user, it is a critical component of the overall FSFN system. Some of the key processes for release 1 include:

• Tickler Escalation;
• Approval History Processing;
• Case Closure;
• Case Part Removal;

The programs and processes in FSFN are designed to run hands-off. The batch scheduler will run on the Autosys which can automatically run scheduled batch procedures, backup procedures, and interface programs.

The batch processes consist of individual Java programs that are arranged in logical groups called run cycles and run inventories. Release 1 batch cycles are documented in the FSFN operations guide.

**FSFN Security Architecture**

FSFN security is a series of interrelated components. Together, they form the building blocks needed to provide a core capability for securing an application from the trust perspective.
**FSFN Authentication through LDAP**

To authenticate FSFN users, the application authenticates users by communicating with DCF's LDAP server. LDAP in conjunction with Tivoli Identity Manager makes up the bulk of authentication processing in FSFN.

- Users are assigned LDAP IDs by DCF.
- A single user may have multiple roles within the FSFN system, but they are all treated as being under the umbrella of a single identity.
- User visits the FSFN login screen and enters user name and password.
- FSFN authenticates using TDS/LDAP.
**Tivoli Identity Server**

- Centralized control for DCF.
  - Ability to revoke access from a single place
  - Centrally set password policy.

- Multiple applications use TDS, and for all of them, there is a single user name and password. When user names and passwords change, they change for all applications using TDS.

- Users can use Tivoli Identity Manager, TIM, to self-service their own account. For example, a user can request a password reset if he forgets his password or locks himself out because of too many failed login attempts.
FSFN Authorization

In FSFN every transaction is secured by an internal security manager. At the lowest level, an application transaction is made up of a struts Path and Action. Resources (Struts paths and actions) are linked to security profiles, security profiles are linked to security groups, and a security group is linked to a worker. All the security related tables are prefixed with SCRTY.
Security Tables in FSFN

**SCRTY_GRP**
This contains information that describes the security group. This information is **populated by the application** via the Maintain Security Groups window.

**SCRTY_GRP_PRFL**
This table links security groups to profiles. Information in this table is **populated by the application** via the Maintain User Groups window.

**SCRTY_JSP**
Topic name or grouping for security resources. This Information is **populated by the developer** and cannot be modified from the system.

**SCRTY_PRFL**
This table contains information that describes a security profile. A security profile contains security resources. This table is **populated by the application** from the Maintain Security Profiles window.

**SCRTY_PRFL_RSRC**
This table links security profiles to security resources. This table is **populated by the application** from the Maintain Security Profiles window.

**SCRTY_RSRC**
This table contains the actual information that a user is authorized against. For FSFN, each transaction contains a path and action. This table is **populated by the developer** and cannot be modified by the application.

## FSFN Security Architecture Summary

<table>
<thead>
<tr>
<th>Security Component</th>
<th>FSFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification &amp; Authentication</td>
<td>FSFN uses LDAP authentication via a user-id/password. DCF policy will drive the usage controls on these user-id’s – password standards, number of invalid logon attempts, time before password is required to be changed, defined period of time before terminating a users session due to inactivity. The maintenance of this information is accomplished through Tivoli’s Security Identity Manager. It provides a secure, automated and policy-based user management solution that helps effectively manage user accounts, access permissions and passwords from creation to termination across the department.</td>
</tr>
<tr>
<td>Authorization/Access Control</td>
<td>Authorization and access control will be supported by the SACWIS application. The CGI-AMS SACWIS solution supports a hierarchical role-based security approach with users, groups, and profiles.</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>Secure Socket Layer (SSL) will provide encryption services so that eavesdropping cannot take place.</td>
</tr>
<tr>
<td>Auditing/Monitoring</td>
<td>Auditing and monitoring services are built into the application. For example, all records are tagged with the user and time the record was created as well who last modified the record. There are also other history tables that are maintained by the system in the event information needs to be tracked and attributed.</td>
</tr>
</tbody>
</table>
**FSFN Release 1 Environments, High Level Overview**

FSFN Release 1 deployment consists of the on-line application environment, the batch and reporting environments, and the interface environment.

**On-line Application Environment**

The FSFN on-line application for Release 1 consists of four tiers as illustrated in the following diagram.

**FSFN On-line Environment for Release 1**

---

**Client Tier**

The FSFN Release client tier consists of the following components:

- Microsoft Internet Explorer
- Microsoft Word 97 through Word 2003
- Microsoft Excel
- Java Script
- ActiveX components
- Word templates

Application Tier
The FSFN Release 1 on-line application server environment consists of two Linux 4 servers in a clustered configuration with a RadWare network appliance balancing the workload with session affinity and failover support. Each server hosts eight BEA WebLogic 9.2 J2EE application server instances. The sixteen application server instances are configured as a single cluster with session replication to support failover.

Database Tier
The FSFN Release 1 database is hosted on a z/OS DB2 8.2 instance. The application tier attaches directly to the database using a pure Java type 4 JDBC driver.

Reporting Tier
The FSFN Release 1 reporting tier consists of a cluster of Business Objects Enterprise XI R2, BOE servers. The FSFN application is integrated with the BOE web based infoView platform through a single sign-on.

All reports in FSFN Release 1 are scheduled under the BOE report scheduler and publication subsystems and produce static output as either a PDF document or an Excel document. No reports are executed on-demand in the BOE environment. The BOE infoView environment is used only to list available reports and historic instances of reports and display the selected PDF or Excel documents.

The reporting tier interacts with three DB2 UDB 8.2 databases. Two of these databases are proprietary Business Objects databases:
- The BOE CMS repository which hosts the object definitions for the BOE Universes, users, reports, and folders;
- The BOE audit database where BOE records key events such as the creation users, reports, folders, …

The third DB2 UDB 8.2 database used by the FSFN reporting tier is the FSFN data marts database. This database contains FSFN data extracted and loaded into the data marts in a nightly cycle by the incremental ETL processes described in the next section. This database is structured differently from the on-line FSFN database and is designed to meet reporting needs.

Batch and Reporting Environment
FSFN Release 1 batch processes utilize three technologies. The first two are managed by the DCF mid-tier job scheduler, CA AutoSys. These processes are:
- Java batch processes hosted on Windows Enterprise server.
o Java batch processes interact with the on-line database and with the FSFN data warehouse through JDBC. In addition a batch synchronization process interacts with the BOE environment to manage user account profiles through the BOE Java API. Each process is structured to manage state in a way that facilitates checkpoint restart capabilities.

- Business Objects Data Integrator, BODI, processes hosted on Windows Enterprise servers.
  
o BODI processes build and manage the FSFN data warehouse databases. These processes extract changes from the on-line database and apply them to the data warehouse databases on a nightly cycle. The processes are structured to run with auto-correct load to facilitate checkpoint restart capabilities.
  
o BODI processes build the report databases as well as the CBC extract files and other FSFN Release 1 interfaces.
  
o On successful completion of the BODI processes that build the reporting database the BODI process generates the trigger file to release the FSFN batch report cycle in the BOE environment.
FSFN Report ETL Environment

![Diagram of FSFN Report ETL Environment]

FSFN Data Warehouse and CBC Extract Data Mart ETL Environment

![Diagram of FSFN Data Warehouse and CBC Extract Data Mart ETL Environment]

CBC FTP Delivery Environment
The FSFN batch report cycle is triggered by the creation of a trigger event file under the control of CA AutoSys. However, the report schedule is managed by the BOE job scheduler which manages the report schedule as well publication / subscription based report bursting. All FSFN Release1 reports generate either PDF files or Excel files as their output format. The reports that generate PDF output are built with Crystal Report. The reports that generate Excel files are built with Deski.

**FSFN Report Production and Distribution Environment**

**Interface Environment**

The diagram below shows the interfaces points for FSFN Release 1. There are three technologies employed to support the Release 1 interfaces: HTTP; FTP; and ODBC.

- HTTP
- MCTS
- Phoenix
- CI Secure document
- Client Photo

- Batch file exchange over FTP
  - CBC Extracts
  - Enterprise Client indexes
  - ICWSIS
  - Department of Juvenile Justice

- ODBC
  - DCF Dashboard SQL Server

Figure 1 presents a high level view of the FSFN Release 1 Interfaces.
**FSFN Release 2A Environments, High Level Overview**

FSFN Release 2A deployment consists of the on-line application environment, the batch and reporting environment, and the interface environment.

**On-line Application Environment**

The FSFN on-line application for Release 2A consists of five tiers as illustrated in the following diagram.
Client Tier

The FSFN Release client tier consists of the following components:

- Microsoft Internet Explorer
- Microsoft Word 97 through Word 2003
- Microsoft Excel
- Java Script
- ActiveX components
- Word templates
Application Tier

The FSFN Release 2A on-line application server environment consists of two Linux Redhat 5 servers in a clustered configuration with a RadWare network appliance balancing the workload with session affinity and failover support. Each server hosts eight BEA WebLogic 9.2.3 J2EE application server instances. The sixteen application server instances are configured as a single cluster with session replication to support failover.

Background Tier

With Release 2A two JMS message servers were introduced to the architecture to handle background processing. The JMS message servers are configured as a distributed queue spanning a cluster eight WebLogic 9.2.3 application server instances distributed between the two servers. While the background JMS servers are in the same Domain as the application tier servers they are in a different cluster. WebLogic SAF (Store and Forward) agents bridge between the application tier and the JMS queues on the background tier. The diagram which follows provide a logical illustration of the process flow associated with FSFN background processes in Release 2A. Note that the background processes in Release 2A utilize the EJB technology.

- Message Driven Beans
- Session Beans
- CMP Entity Beans

FSFN JMS Audit Log
Database Tier
The FSFN Release 2A database tier consists of a middle-tier component in addition to the z/OS DB2 8.1 database carried over from Release 1. The middle-tier component of the FSFN on-line database is hosted on DB2 UDB 8.2 running on Linux Redhat 5. This database holds Department of Health Birth Record data, image data, and the FSFN user activity audit data.

The application tier attaches directly to both of the FSFN on-line database platforms using a pure Java type 4 JDBC driver.

Reporting Tier
The FSFN Release 2A reporting tier consists of a cluster Objects Enterprise XI R2 servers. As with Release 1, the FSFN application is integrated with the BOE web based infoView platform through a single sign-on. The following changes are made to the FSFN reporting environment in Release 2A.

- To provide more robust production and delivery of reports in FSFN Release 2A a file server was added to the architecture. This enables either server to access report files. In Release 1 the primary BOE server doubled as the file server.
- The CMS database which serves as the repository for Business Objects is migrated to Oracle from DB2 UDB to remedy a performance problem.
- A third server was added to the BOE cluster. This server is used primarily as a work server. It does not host the end-user facing CMS service.
- Reports in Release 2A were delivered as on-demand reports rather than static scheduled reports without output to PDF and Excel. Release 2 A reports execute queries directly against the reporting database when the report is viewed. These reports are developed using Webi and Crystal.
- Power users are granted rights to navigate the FSFN BOE Universe to analyze data and develop and schedule custom report views for distribution.

Batch and Reporting Environment
FSFN Release 2A batch processes utilize the same three technologies utilized for FSFN Release 1. For additional details on the batch environment please refer to the section on the Batch and reporting environment for FSFN Release 1.

FSFN Report Production and Distribution Environment
Interface Environment

The diagram below shows the interfaces points for FSFN Release 2A. There are three technologies employed to support the Release 2A interfaces: HTTP; FTP; and ODBC. Note that in Release 2A the MCTS and Client Photo interfaces are obsolesced because the functionality of those systems was incorporated into FSFN. Two new interfaces were added to FSFN in Release 2A. The first interface replaces the interface between FDLE and MCTS because FSFN takes the place of MCTS. The second interface is a file interface from the Department of Health Birth Records System. Each week the DOH system sends a file via FTP that contains new and updated Florida birth records from the previous week. These are loaded into the FSFN mid-tier on-line database which the application searches.

- HTTP
  - Phoenix
  - CI Secure document
  - FDLE
- Batch file exchange over FTP
  - CBC Extracts
  - DOH Birth Record
  - Enterprise Client Index
  - ICWSIS
- Department of Juvenile Justice
- ODBC
- DCF Dashboard SQL Server

Figure 2 presents a high level view of the FSFN Release 2A Interfaces.

Figure 2: FSFN Release 2A Interfaces

FSFN Release 2B Environments, High Level Overview
FSFN Release 2B deployment consists of the on-line application environment, the batch and reporting environment, and the interface environment.
On-line Application Environment

The FSFN on-line application for Release 2B consists of five tiers as illustrated in the following diagram.

FSFN On-line Environment for Release 2B
Client Tier

The FSFN Release client tier consists of the following components:

- Microsoft Internet Explorer
- Microsoft Word 97 through Word 2003
- Microsoft Excel
- Java Script
- ActiveX components
- Word templates

Application Tier

The FSFN Release 2B on-line application server environment consists of two Linux Redhat 5 servers in a clustered configuration with a RadWare network appliance balancing the workload with session affinity and failover support. Each server hosts eight BEA WebLogic 9.2.3 J2EE application server instances. The sixteen application server instances are configured as a single cluster with session replication to support failover.

Background Tier

With Release 2B the use of the two JMS message servers is extended to handle background interface processing with the FLORIDA on-line IMS system. As with Release 2A, the JMS message servers are configured as a distributed queue spanning a cluster eight WebLogic 9.2.3 application server instances distributed between the two servers. The background JMS servers remain isolated in their own cluster. WebLogic SAF (Store and Forward) agents bridge between the application tier and the JMS queues on the background tier. The diagram which follows provides a logical illustration of the process flow associated with FSFN background processes in Release 2B. The diagram also includes all of the phases of the FLORIDA Interface process:

- FSFN on-line clearance against the FLORIDA client registry – FL ODS
- FSFN batch process to retrieve the results of the application from FL ODS
- FSFN event driven background process submitting eligibility determination to FLORIDA and the FLORIDA Work Manager, AMS WM

FSFN JMS Audit Log
Database Tier

The FSFN Release 2B has no additional databases added from Release 2A and the database engines remain at the same release levels.

Reporting Tier

The FSFN Release 2B reporting tier is built on the same BOE servers at the same releases levels and configuration as for FSFN Release 2A. The FSFN data mart is upgraded to DB2 LUW 9.5 64 bit running under Redhat Linux 5 64 bit. This new database configuration supports increased usage of on-demand reports and on-line analytical processing in the FSFN BOE environment.

Batch and Reporting Environment

FSFN Release 2B batch processes utilize the same three technologies utilized for FSFN Release 1. For additional details on the batch environment please refer to the section on
the Batch and reporting environment for FSFN Release 1. The following changes are made to the FSFN reporting environment in Release 2B.

- The database server hosting the FSFN data warehouse, the FSFN data marts, and the BOE audit database is moved to a new server with increased capacity and the database engine is upgraded from DB2 UDB 8.2 to DB2 LUW 9.5 64 bit on Redhat Linux 5.

**FSFN Report Production and Distribution Environment**

![Database Diagram]

**Interface Environment**

The diagram below shows the interfaces points for FSFN Release 2B. There are five technologies employed to support the Release 2B interfaces:

- Web Services;
- HTTP;
- (S)FTP;
• IMS Connect;
• ODBC.

Note that in Release 2B the MCTS and Client Photo interfaces are obsolesced because the functionality of those systems was incorporated into FSFN. Two new interfaces were added to FSFN in Release 2A. The first interface replaces the interface between FDLE and MCTS because FSFN takes the place of MCTS. The second interface is a file interface from the Department of Health Birth Records System. Each week the DOH system sends a file via FTP that contains new and updated Florida birth records from the previous week. These are loaded into the FSFN mid-tier on-line database which the application searches.

• Web Services
  o Phoenix call record
  o CBC check payment updates
• HTTP
  o Phoenix
  o CI Secure document
  o FDLE
  o AMS WM – FLORIDA’s Work Management system (access to interface PDF documents)
• Batch file exchange over FTP
  o CBC Extracts
  o DOH Birth Record
  o Enterprise Client Index
  o ICWSIS (will be obsolesced on full implementation of Release 2B)
  o Department of Juvenile Justice
  o Adoption web application
• IMS Connect
  o FLORIDA
• ODBC
  o DCF Dashboard SQL Server
  o FL ODS – FLORIDA’s on-line relational database extract
  o AMS WM – FLORIDA’s Work Management system

Figure 3 presents a high level view of the FSFN Release 2B Interfaces.

Figures 4 and 5 present high level views of the two way interface between FSFN and the CBC financial accounting systems. The outbound interface though Secure SFTP delivers
payment documents to the CBC financial accounting systems. The inbound side enables the CBC system to update FSFN with check numbers for the payments.

Figures 6 present high level view of the FSFN-FLODS (FLORIDA) batch interface between FSFN and Florida system.

The architecture for the Remote Data Capture Device interface is presented in more detail in the section following figure 5.

**Figure 3: FSFN Release 2B Interfaces**

**Figure 4: CBC Check Payment Interface**
Figure 5: CBC Financial Extract Interface

Figure 6: FFSN-FLODS batch interface
Remote Data Capture (RDC) Device Server Side Architecture

FSFN is extended using web service technology to support the use of mobile device technology by workers in the field to enable them to leverage FSFN case data on-site during field visits and to capture photographic and form data associated with the visit to be added to the FSFN case record. This section defines the architecture for the FSFN server side extensions that support the use of mobile device technology in support of case work in the field. This section provides details on the following elements of the mobile device support architecture:

- Security;
- Web Service Implementation;
- Web Service Message Implementation
Security

Security for the server side components of the remote data capture solution is built on the FSFN application security infrastructure. This is detailed in the following topics:

- Authentication;
- Authorization;
- Confidentiality;
- Data ownership;
- Auditability.

Authentication:

Authentication is the process for verifying the identity of the users of the remote data capture web services. Only registered FSFN users are permitted to utilize the web services that support the FSFN remote data capture systems. Just as FSFN users must have their user name and password credentials registered in the DCF LDAP directory so must the RDC users. The authentication process for the FSFN RDC web services checks the user name and password for each web service call against the DCF LDAP directory.

If the User name and password combination do not match a valid active account in the DCF LDAP directory, the user is not permitted to use the RDC web services.

Authentication is performed through Basic Authentication supported by the Weblogic application server. This approach assumes that the mobile client application will have coded the authorization header for the web service call with the user name and password for the mobile user. This requires the configuration of the DCF LDAP directory service as a security provider for the Weblogic environment. These configuration settings are detailed in the authorization section below.

Authorization:

Authentication is the process for verifying that an identified user has the rights to perform the requested action. For the FSFN RDC web services the DCF LDAP repository is used to determine whether the user is an authorized FSFN user. Each FSFN user is associated with the LDAP group FSFN_USER. This group authorizes the user to access the FSFN application as well as the FSFN RDC web services. This is implemented similar to the FSFN online authentication. Each web service call will go through the LDAP authentication. Each web service requires a header form which contains the LDAP username and password attached to it, which will be used to authenticate the user against LDAP. The username provided should be a valid LDAP account under the user group FSFN_USER. LDAP accounts that belong to different user groups won't be valid. A valid LDAP user account under the FSFN_USER user group is REQUIRED in order for the web service to go through. If the username passed with the web service is not a valid LDAP account under the FSFN_USER group or the password is wrong or expired the web service will fail to execute and it throws the below error codes and error messages.
Confidentiality:
Confidentiality means that the data is protected from authorized viewing while it is in transit between the FSFN RDC client systems and the FSFN server. The FSFN server side application is accessible only within the DCF intranet. It is assumed that the FSFN RDC client applications will access the FSFN servers through the DCF Aventail gateway which provides a secure / encrypted virtual private network to prevent sensitive information from unauthorized access while it is in transit over the internet.

Data ownership:
In FSFN a user must have an assignment to a case in order to add or make changes to the case record. The FSFN RDC web services enforce this same data ownership constraint.

Auditability:
All FSFN RDC web service calls are logged to the same audit log tables as on-line request are logged to. The audit log details for FSFN RDC web services are recorded in XML format just as the on-line audit log. The audit log details for FSFN RDC web services are include:
- What worker role executed the web service;
- What web service function was executed;
- When the web service was executed;
- The content of both the request and the response for the web service.

Web Service Implementation
The FSFN web services are defined using Java EE 5 annotations under the W3C Web Service Policy Framework, WS-Policy. The following annotation and referenced file content define the service implementation:

```java
@Policies({
    @Policy(uri="FsfnWsAuth.xml", direction=Policy.Direction.inbound)
})
```

Network configuration:
There are no changes required to the FSFN network configuration to support the FSFN RDC web services. The FSFN RDC web services are deployed on the same WebLogic application servers as the FSFN application and the existing FSFN web service interfaces. Accordingly, they are accessed through the same RADWARE front end URL as the FSFN application and the existing web services supporting Phoenix and CBC financials. fsfn-production.dcf.state.fl.us will be the production URL.

Installation and Configuration Architecture:
The FSFN RDC web services will be deployed using the standard automated FSFN build deployment processes. No changes are required to the build process to support the deployment of the FSFN RDC web services. Aside from the security configuration settings defined in the security section of this document above there are no configuration changes required to the FSFN servers to support deployment of the FSFN RDC web services.

FsfnWsAuth.xml referenced above
<?xml version="1.0"?>
xmlns:wssp="http://www.bea.com/wls90/security/policy">
<wssp:Identity/>
</wsp:Policy>

Web Service Message Implementation

FSFN RDC web service data is exchanged using XML strings. The functional design specification for the FSFN RDC web services includes XML schemas defining the structure of the input and output messages for each of the web service calls supported.

The XML strings are converted to and from the Java objects used in the business logic tier using JAXB with JSR 173 StAX (Streaming API for XML). The following jar file are included in FSFN to support this infrastructure:

- jaxb-impl.jar
- jaxb1-impl.jar
- jaxb-api.jar
- jaxb-xjc.jar
- jaxrpc.jar
- jsr173_1.0_api.jar

Error Message Architecture:

With the exception of application errors the FSFN RDC web services will rely on the standard HTTP web service error messages. Application error messages will be returned as part of the standard FSFN RDC web service message structure defined in the FSFN RDC web service functional documentation.

None functional exceptions such as server unavailable, timeout, authentication failure, authorization failure are handled by the web service infrastructure and return standard HTTP error codes.