Family Automated Client Tracking System

System Architecture Specification

October 22, 2007
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.21 ITS Firewall Standards</td>
<td>188</td>
</tr>
<tr>
<td>12.22 Supporting Documentation for Technical Architecture Components</td>
<td>189</td>
</tr>
<tr>
<td>12.23 Information Technology Change Control Committee Charter</td>
<td>190</td>
</tr>
<tr>
<td>12.24 Project Plan – Phase 2 – System Development Phase Plan</td>
<td>193</td>
</tr>
<tr>
<td>12.25 System Design Documents</td>
<td>194</td>
</tr>
<tr>
<td>12.25 Technology Architecture – Phase I FACTS</td>
<td>195</td>
</tr>
<tr>
<td>13 Revision History</td>
<td>196</td>
</tr>
</tbody>
</table>
1 Introduction
Design and development of information systems that are enterprise wide in scope require a solid technical foundation (i.e. the software, network, hardware and database platforms). This FACTS document entitled System Architecture Specification (SAS) addresses those key technical elements, defining what these building blocks are, why they were selected and documenting the assumptions and constraints that were made in the decision process.

1.1 Purpose of this Document
This document serves as a ‘roadmap’ for CYFD Information Technical Services (ITS) as it embarks upon the many technical avenues/decisions that must be considered in developing a web-enabled, Service Oriented Architecture (SOA) and federally compliant Statewide Automated Child Welfare Information System (SACWIS); or better known to the State as the CYFD Family Automated Client Tracking System (FACTS). This document also serves to provide a record of the decision making processes and the resulting determination of technical choices/strategies that were made regarding the overall design, development and implementation of FACTS. This document serves to define the comprehensive framework that describes the functional, application and technical architectures that comprise any complete system.

The FACTS SAS will focus on the governance, technical (application and technical) architecture as the business (functional) architecture will be presented more thoroughly in the System Requirements Specification – and also documented in ‘Topic Papers’. The technical architecture is further divided into hardware, network, data and software components required to deliver an information system's functionality. The software component will specify the requirements for the middleware that are the key technological components, the ‘plumbing’ for a SOA. These components will incorporate the technologies, products, standards and interfaces required to build and operate FACTS. The System Architecture Specification (SAS) must provide continuity for the life of the project, aid in the transition from one stage to the next, and provide each new team with the technology background required to produce results quickly. The technical teams for each stage are expected to maintain the SAS, keeping it current as the project progresses.

The heart of the SAS shows the network components (intranet and internet) over which FACTS will be distributed, the hardware required for its operation, the underlying implementation software, and the data used or manipulated by the system. However, these views are insufficient to provide the context and background necessary for design and development teams as they begin a new stage of the project. Therefore, the System Architecture Specification contains additional views and combinations of views such as:

- Environment Models that show what software will be located on which hardware,
- Interface Models to identify how FACTS will interact with other systems, and
- Security Models showing how privacy and information will be protected.
Each architecture is represented in the SAS by one or more models. A model is a representation of the real world; it can be a description in words (in paragraphs, bullet points or tables) or it can be a graphical representation. Often, both types of models are employed to provide the greatest understanding. In addition to providing various views, the SAS must provide the context for them. This context includes:

- the standards and preferences selected or developed by CYFD and ITS,
- the current and projected state of technology,
- the important decisions that have been made for FACTS,
- the information that supports those decisions,
- the approach or methodology for building the models and architectures, and risks inherent in the decisions and solutions, and mitigating factors or strategies.

The context also includes working documents or other artifacts generated during the course of architecture development. These documents and artifacts are not part of the architecture, but will be included in the SAS either as reference points to document the approach taken to developing the architecture, or as an Appendix.

Understanding the context of the architecture decision will assist ITS in several ways to complete the FACTS implementation. ITS can take advantage of infrastructure, standards, processes, and support mechanisms already in use. It will aid in understanding factors included in the decision and to be aware of the importance of these factors so that system integrity is maintained as FACTS is modified and extended. It will also help to identify what was NOT included in the decision, so that if new factors become evident there will be no artificial barriers to changes.

Finally, this document provides models, and information about an operational FACTS and the network on which it is projected to run – the Department of Information Technology/General Services Department (DoIT/GSD) Wide Area Backbone. These models will contribute toward development of a system satisfactory for the citizens of New Mexico and aid in improving productivity of ITS personnel. ITS can anticipate and identify potential performance degradation or bottlenecks. These models allow for the acquisition of the right components for FACTS to mitigate or resolve issues and will be developed to be used, reused and enhanced as the project progresses.

The SAS provides the underlying structure and framework upon which the application is built and executes and information about an operational FACTS and the network on which it is projected to run. However, it is independent of the application itself. This independence allows for flexibility in designing and building the application and allows the architecture to be used for other applications with minimal change. The technology specified in the architecture must allow the application to deliver all required functionality; however, defining and documenting the application functionality is not one of the SAS objectives. The techniques used to keep the technical architecture in compliance with the application requirements is documented in Section 1.6: “Approach to FACTS Architecture Development.”
This document is based on the State of New Mexico’s Office of Chief Information Office (OCIO) Best Practices approach (garnered from the National Association of the State Chief Information Offices (NASCIO)) in defining System Architectures currently specified as the Information Technology Enterprise Architecture (ITEA) approach. A system architecture is the comprehensive framework that describes a system’s functionality, application components, along with the software, database and hardware and network components used to deliver that functionality. A system architecture consists of seven architectural components, as shown in Figure 1.2-1 and described below.

The business (functional) architecture describes the conceptual view, or what the planned system will do. It shows the breakdown of the system solution into functional areas and how they interact. Functionality can be described using models such as web services, use cases, object models (including functional area models), logical data models, and user interface models. Through these techniques, the system scope is established, communicated to stakeholders, and validated against business objectives and operational performance targets. All of these models will be found in the following documents:

- **System Requirements Definition**: high-level use cases
- **Conceptual Design Documents**: expanded use cases, object models, logical data models, user interface models
- **Detail Design Documents**: expanded use cases, object models, user interface models

There are many modeling techniques to define the business architecture. The common denominator is that they all strive to capture what business functions will be automated, rather than showing how the system will physically be implemented. A business architecture generally starts with a context diagram of the proposed system.

The actual implementation of the business architecture is defined through the other six components of the system architecture: architecture blueprint, the application, software, data, hardware and network and governance architectures. These six describe how the business architecture will be implemented physically. They will be described in more detail in the sections dedicated to their descriptions. They are, however, defined briefly below.
Figure 1.2-1 Seven Components of System Architecture

- The architecture blueprint describes the business drivers for the Agency – those initiatives promulgated by the Governor and Senior Staff that IT needs to address to make IT responsive to their business needs.

- The governance framework describes the policies, procedures that help management control IT resources throughout the software development lifecycle (SDLC).

- The application architecture describes the partitioning of the application, the interface management and security processing.
• The software architecture describes the software layers and components and how they interact to implement the functional architecture.

• The data architecture addresses the physical design, structure, and implementation of the persistent (not transient) data needed for the planned system’s functionality and operations.

• The hardware and network architecture describes the computers and networks on which the system operates and their configurations.

The seven component architectures are the primary building blocks, but they cannot be defined in isolation. Each affects the definition of the others. A system’s architecture must articulate how these components are related and will interact. This is further complicated because even the smallest system involves multiple versions of these component architectures to address the different environments needed to create and maintain a system, in addition to implementing it in production.

Moreover, the system architecture must make these relationships and interactions understandable - at an appropriate level of detail - to State stakeholders, the development team and operational staff alike. The software, data, hardware and network architectures are often collectively called the technical architecture. Figure 1.2-2 shows the relationship between the building blocks.

*It should be noted that since the Business Architecture is inextricably tied into the specification of user requirements, it will be documented in the System Requirements Definition (SRD); and Topic Papers. It will also be documented in the Conceptual System Design (CSD) and the Detailed System Design (DSD) and will, therefore, not be discussed in any detail in this document.*

This is not a static document. It will be updated whenever the technical team garners new architectural information.
Figure 1.2-2 also highlights the initial phases of implementing the Best Practices Methodology for defining System Architecture. As mentioned above, the Functional Architecture is documented in the SRD and will not be covered in any detail in this document.

The Application Architecture is the first system component that was analyzed. It is only when the complete design has been specified that it is possible to fully decide what hardware/network and database platforms are appropriate for implementing the user application.
In addition to CYFD Advanced Planning Document (APD), Requirements Definition Sessions and Technical Meetings were conducted with the State stakeholders. At these meetings the overall technical assessment process and technical requirements were discussed. (Similar types of meetings will be conducted when we embark on the development of FACTS.) The specific decisions and processes will be discussed in more detail in the architecture sections that apply. The following, however, delineates the topics discussed:

- Validate Technical Requirements,
- Examine Existing Infrastructure of the State Network,
- Assess Application Architecture,
- Weighting the Architecture Evaluation Criteria,
- Evaluate Single vs. Multi Vendor Application Development Environment (ADE),
- Evaluate Thin vs. Ultra-Thin Clients, and
- Evaluate Centralized vs. Distributed Application Architecture.

There were also workgroups established to analyze and select a security model, an application development environment, and middleware.

See Figure 1.2.3 for a graphic of the State of New Mexico’s Enterprise Architecture Specification.
SSA – One of the Business Domains in the Business Architecture component of the NMEA

Highest occurring technical domains seen at other states.

Governance
- EA Processes
  Governance Structure
- Quality Program Performance Measures
- Procurement Processes
- Project Management Processes

Based on Existing Docs such as Governor Performance review

Do we adopt the Federal e-gov architecture -(will influence future vision, technical model, standards)- Principles

Governance Structure

Do we adopt Federal Business Reference Model- will influence definitions and inventory of Lines of Business Principles

Social Services Architecture Best Practices Processes
Social Services Architecture Best Practices
Social Services Architecture
Social Services
Social

Justice Architecture Best Practices Processes
Justice Architecture Best Practices
Justice Architecture
Justice

Resource Mgmt Architecture Best Practices Processes
Resource Mgmt Architecture Best Practices
Resource Mgmt Architecture
Resource Mgmt

Education Architecture Best Practices Processes
Education Architecture Best Practices
Education Architecture
Education

Government Operations Architecture Best Practices Processes
Government Operations Architecture Best Practices
Government Operations Architecture
Government Operations

Data Mgmt BP Stds Inventory
Data Mgmt BP Standards Inventory
Data Mgmt

Applications BP Stds Inventory
Applications BP Standards Inventory
Applications

Middleware Inventory
Middleware Inventory
Middleware

Info Protection/ Security BP Inventory
Info Protection/ Security BP Inventory
Info Protection/ Security

Network BP Inventory
Network BP Inventory
Network

System Mgmt BP Stds Inventory
System Mgmt BP Standards Inventory
System Mgmt

Enterprise Architecture Guiding Principles

Business Architecture

Technical Architecture

Business Domains

Technical Domains

Architecture Blueprint

Trend and Drivers

Business Goals

Objectives

Strategies
1.3 Background for FACTS Architecture

Child welfare services in CYFD are delivered in a state-supervised environment. ITS is responsible for the supervision of all programs designed to ensure that child welfare services are delivered in accordance with Federal and State law. The actual delivery of child welfare services is performed by CYFD employees housed in county or regional offices. CYFD has determined that child welfare data must be collected in an integrated fashion. Specifically, the information must be inclusive from the first contact with the child and his or her family through the time that the child is no longer involved with the child welfare system. This includes the provision of investigative services, pre-placement, preservation, preventive services, placement services, completion of reunification services, adoption or the provision of independent living services to those children who cannot be reunified or adopted. Certain activities must be recorded over the period of involvement in order to assess the impact of intervention and program goals.

Federal legislation, PL 96-272, requires states to implement an information system to track children in substitute care and to establish other child protections to qualify for enhanced Title IV-B funding. These and other federal dollars support the child welfare services being provided today to CYFD children.

Over the past few years, there has been persistent promulgating of legislation directed at the development of programs that address the burgeoning needs of abused and neglected children and their families. These needs have been heightened by the increased demands placed upon the child welfare system as a result of personal responsibility expectations, family instability and more devastating nature of family problems, as well as more general societal problems such as unemployment, poverty and violence. As technology has become more robust and affordable the development and implementation of the Statewide Automated Child Welfare Information Systems (SACWIS) to address these needs became not only feasible but warranted.

A by-product of the various child welfare legislation enacted during this period has been the development and implementation of information systems in many states that satisfy the categorical information needs of related programs, rather than just the cursory needs of the clients and human services staff who are directly involved in the child welfare service delivery system. Programs such as child protective services, family preservation, foster care, and adoption, as well as numerous ancillary children programs, each have their own informational requirements. In addition, other financial resource programs add major and unique requirements for gathering information regarding eligibility and for the reporting of varying aspects of that information to the sponsoring agencies. Beyond traditional Title IV-B, Title IV-E and Social Services Block Grant (SSBG) these programs include Temporary Aid for Needy Families (TANF), Medicaid, Child Support Enforcement, and various others.

SACWIS was the first Federal legislative initiative that provided resources for the merging and linking of these multiple information requirements into a single child welfare services focused system. The Omnibus Budget Reconciliation Act of 1993 (P.L.
103-66), Section 13713 made funding available for the planning, design, development and installation of statewide automated child welfare information systems.

For a limited window of time, October 1993 through September 1997, most States were able to design integrated child welfare information systems utilizing Federal resources specifically earmarked for this purpose. At this point most states have or are in the process of implementing state-of-the-art systems.

We should follow the principles and purpose of an Information Technology architecture as defined by the Meta Group, a prominent IT consulting firm:

“An architecture:

- must use technology to enable and encourage rapid process change;
- must provide a framework for coherent system design and technology selection;
- must maximize the ability to leverage existing technology assets;
- must enable inter/extra agency communications and information sharing;
- must recognize that technology in and of itself does not solve business problems and that only an architecture that is doable, flexible and changeable can provide a workable value-added solution.”

The content of the SAS was first established by the creation of an outline that was reviewed and commented upon by members of the ITS Technical Team.

1.4 **Overview of Document Contents**

This document starts with introductory comments on its purpose, scope and background.

Section 1 also specifies the objectives of an information technology architecture, the approach in developing the architecture and the intended audience for this document.

Section 2 provides a system overview, specifying the purpose of FACTS, its architecture, intended use and operation and its planned deployment.

Section 3 specifies the critical success factors, design decisions, assumptions and constraints while developing this document. Documenting decisions and their driving factors will help ITS and the implementation team make better decisions as FACTS progresses.

Section 4 provides an overview of the business architecture and its relationships with the application, hardware, network and software, and data architectures. Business Architecture is not detailed in this document, however, since it is covered in the Systems Requirements Definition, Conceptual System Design, and Detailed System Design documents.
Section 5 starts the process of documenting the various components of the system architecture. It details the application architecture, its description, and the approach used. It then defines the partitioning of the application, the interfaces with other systems and users and the security model to be implemented. The conclusion describes a discussion of the decisions, assumptions and constraints considered as the application architecture was defined.

Section 6 continues with a discussion of the hardware and network architecture. It provides a description and the approach used in developing it. This section details the specific hardware and network elements that were selected to implement this architecture. It concludes with a discussion of the decisions, assumptions and constraints that were made while defining this component.

Section 7 specifies the software architecture: the operating system, database management system and ancillary software used to support the application. It also provides a description and the approach used in developing the software architecture. It concludes with a discussion of the decisions, assumptions and constraints that were made.

Section 8 specifies the last component of the system architecture, the data architecture. This section provides a description and the approach used in developing it. The database management system characteristics that were selected are also described. There is also a discussion on the recommended Data Warehouse approach. The next section provides information on the entity relationship diagrams with a description of the tool used with the corresponding semantics defined. The section concludes with a discussion about data retention issues.

Section 9 describes Governance, the foundation upon which all of information technology is built upon. It defines: what is overall governance, IT governance, and finally the new world of SOA governance.

Section 10 describes the Disaster Recovery Plan. A Guideline is provided to accentuate the key elements of a plan that is recommended. There is a well documented plan for the current FACTS system.

Section 11 describes the different environments needed to develop, test and implement FACTS. These include development, system test, user test, and user training and production environments. As we migrate to an SOA environment, the complexity of testing new services will cause us to re-visit this environmental model.

**1.5 Objectives of an IT Architecture**

Information Technology (IT) Architectures have tremendous leverage in helping enterprises perform their missions. IT Architectures benefit enterprises by addressing the three vectors driving service delivery systems: speed and flexibility, quality of systems and processes, and cost containment. Because IT is so pervasive, these benefits accrue to the entire enterprise, not merely to the IT department or to islands
within the enterprise. The following three sections are ways in which IT Architectures address each of these drivers.

1.5.1 Speed and Flexibility
The technologies and products an organization or project uses define what is often called its technology footprint. The architecture, when respected, ensures that the footprint is wide enough to allow everything the organization must do to support business needs, but not so wide that the organization is constrained by the complexity and cost of integrating disparate and potentially incompatible components. The time involved in dealing with these integration complexities can be much greater than the time required to build or to acquire and install the initial solution. By helping to cut down on integration complexities and the time it takes to deal with them, and by promoting the reuse of application code via business services, the architecture can help an enterprise to respond more quickly to changing conditions and to the needs of the Agency’s clients, providers, and staff stakeholders.

ITS has chosen to implement an n-tier architecture for FACTS. Section 5: FACTS Application Architecture describes how an n-tier architecture supports flexibility by separating a software application into tiers or layers that can remain unaffected when changes are made to other layers. It should also be noted that the application will be developed on a platform imbedded in the Services Oriented Architecture (SOA) paradigm and implemented in a Web Services environment.

1.5.2 Quality of Systems and Processes
Building computer applications under the guidance of an IT architecture promotes the use of products and components already available within the organization or even within a particular project. If new products are required, the IT architecture helps select new components that work well with existing ones. Whether purchased or built, a component already in production is very likely to be both stable and understood, and well supported. Therefore, the system built with existing components where possible and where the component conforms to the requirement of the new system is likely to be of higher quality than if built entirely new, and computer operations and support staffs are much more likely to be able to respond quickly and correctly to events involving those components. The net result is better service for the IT customer.

Using existing components and a consistent set of tools allows development and support staff to be more versatile and flexible; they do not need to learn about new tool sets or operational procedures if they change assignments. Methodologies, processes and best practices are easier to assimilate into the organization, and can be the same from team to team and project to project.

1.5.3 Cost Containment
Implementations of application architectures contain costs by promoting software reuse in two ways and by reducing the ongoing costs associated with support organizations. The greatest cost savings from the implementation of an application architecture is achieved by eliminating the need to build or buy a software infrastructure because the
need can be fulfilled by something internally, something the enterprise already has developed. Working with a single consistent infrastructure assists application developers in learning and adhering to best practices, since what they learn can be applied to many different application development projects. This reduces training costs and increases the flexibility of people within the organization, allowing it to do more with the people it already has. In addition, the best practices can be tailored to promote building systems that are flexible and easier to maintain and operate. Because the infrastructure built with products designed to integrate readily with each other, this also contains the costs associated with responding to and debugging the inevitable conflicts between products on the desktop or server; keeping redundant, ‘shadow’ support organizations within the business offices for products unsupported or under-supported by the Information Technology department.

Smaller cost savings are generated from the help desk and technical organizations to support additional applications. The cost is reduced to hire and train people to support multiple applications that use similar processes to accomplish application tasks.

Software reuse is another way for application architectures to contain costs. This is accomplished in two separate mechanisms. First, by having a maintainable framework of common application functions, the architecture helps avoid maintaining duplicated functionality in multiple places. Additionally, the framework provides an environment designed for integration so interfacing with existing systems can be accomplished in varied ways with less to no impact on the presentation methods.

1.6 Approach to FACTS Architecture Development

This section provides an overview of the System Architecture definition and selection process. Each of the component architectures is described in greater detail in its own separate section.

The FACTS System Architecture was developed using the American Management Systems’ Best Practices approach to defining System Architectures. We will, however, enhance this approach in specifying the FACTS System and incorporate the ITEA Methodology adopted by the State Office of the CIO. At a high level, these approaches have two steps: first define the component architectures; then show additional views combining the component architectures in various ways.

1.6.1 Component Architectures

The component architectures are the basic building blocks of the System Architecture. Figure 1.2-1: Seven Components of System Architecture illustrates and summarizes these components.

1.6.2 System Architecture Validation Process

The process of developing or validating the system architecture is:

- Review the technical requirements in the context of the enterprise,
Select or define the Application Architecture,
Select the application development tools and environment, and
Define the remaining technical architecture component architectures.

1.6.2.1 Review Technical Requirements

The System Architecture is built to support the planned application within the context of
the enterprise; not only must it enable the application, it must do so in a way that fits the
constraints and culture of the enterprise, allowing ITS to be flexible and contain costs.

To ensure that the System Architecture will support all of the functional requirements of
the application, the available information about the functionality of the proposed system
was reviewed. Available information was extracted from the CYFD Advanced Planning
Document (APD). This information is documented in the System Requirements
Specification. This review process identified requirements that may be significant or
needs special attention such as unusual hardware, activities that take place in hard-to-
reach locations, special data access, etc. This activity is not scientific. What is
“significant” depends upon the business domain, the current technical environment, and
the extent to which the current environment differs from the target application
environment. See Appendix 12.13, Current Technical Environment Overview. This
process relies on the experience of the analysts and their ability to understand the
implications of the requirements.

1.6.2.2 Select Application Architecture

The Application Architecture is defined as partitioning, interface analysis and security
modeling, and is done independently of the tools or other hardware and software.
However, it may determine, or at least has a significant impact on, the tools selected
and on the rest of the technical architecture. Section 5: FACTS Application Architecture
describes this process in greater detail, and shows the results of the selection and
definition process.

1.6.2.3 Select Application Development Environment and Tools

Commercially Off-The-Shelf (COTS) software development tools or open source
development tools will be used in the design/development environment to develop the
FACTS application. These development tools must be commercially available and
include warranty and software support. We are currently working with NetBeans as our
independent development environment (IDE).

The selection of the Application Development Environment (ADE) begins by looking to
the application requirements and is driven to support all the functionality they describe.
This is accomplished through a quantitative analysis of all the application architectures
fit for the individual requirements. See Section 5.6 where the process has been
described in detail along with the additional areas of technology that was discussed in
conjunction with selecting the architecture.
After the application architecture has been selected, the investigation into the capabilities required for the ADE to fulfil the application requirements began. This was conducted through a series of presentations and questionnaires requesting information from the major vendors delivering ADE tools. See Sections 5.3.1, 5.3.3-5.3.5, 5.5, 7.3, 7.7 and 7.8 for Capability Requirements for the ADE. From these inputs, a Capability Matrix was developed to detail the necessary capabilities an ADE tool must contain to be used for the development and implementation of FACTS.

1.6.2.4 Define Remaining Component Architectures

Once the Application Architecture decisions affecting the Technical Architecture are made and development tools are selected, the Hardware and Network Architectures, Software Architectures, and Data Architectures can be completed. Sequence is not particularly important in this step; in fact, the interrelationships between the components largely assure that these steps will not be completed in any specific order.

Usually there will be many known and agreed-to components, but there are also likely to be areas of ambiguity. Where decisions must be made, the requirements gathered in developing the Application Architecture are used as input for a product selection process. Selecting products fills in architectural gaps and modifies the FACTS architecture.

Once the hardware, network and software components have been identified, the network topology can be completed and the various environments (development, system test, user test, user training and production) can be documented.

1.7 Intended Audience

The intended audience for the SAS is:

- ITS Management staff,
- Steering Committee,
- ITS Information Technology staff,
- Federal Reviewers and Auditors (ACF),
- Quality Assurance vendor, (Independent Validation and Verification- IV&V)
- Software designers and developers,
- ITS Operations and Maintenance staff (for post deployment)
- Other State Agencies who may want to review this document
2 System Overview
This section gives an overview of the CYFD SACWIS System that will be used in Requirements Definition Sessions to flush out the requirements for the FACTS. This is not the system in terms of the architectures but the business processing accomplished by the final product.

2.1 FACTS Purpose
The CYFD Family Automated Client Tracking System (FACTS) will accomplish at a minimum the following:

- Collect client data from initial contact through case closure,
- Meet reporting requirements of AFCARS, NCANDS and federally assisted programs,
- Support efficient, economical and effective program administration,
- Provide accurate, up-to-date client and provider tracking,
- Produce management reports with descriptive, statistical and trend data,
- Provide workers with efficient mechanisms for casework documentation, and tools that guide practice, and
- Generate necessary payments, accounting and budget processing.

Figure 2.2-1 provides a graphic representation of the Business Architecture of FACTS’s Baseline SACWIS System.

2.2 FACTS General Nature and Architecture
The FACTS Baseline SACWIS System design’s core modules and structure correspond to the fundamental business processes of child welfare. These include:

- Common Applications Functions
- Staff Management;
- Service Management
- Provider Management;
- Financial Management; and
- Interfaces

See Figures 2.2-2 through 2.2-8:
Figure 2.2-1: Functional Architecture of AMS's Baseline SACWIS System
The SACWIS design incorporates a single statewide database that records person and event information. The person and event based approach is a logical way to organize child welfare information. While families and cases change their composition over time, persons and events, once described or documented, do not change.

Figure 2.2-2 CYFD FACTS
An additional support module made up of Common Application Functions (CAFs) forms the application infrastructure and includes functions such as On-line Help and Policy, Ticklers, Document Management, and Automated Messaging.
The Staff Management module maintains information about staff demographics, location, assignments and workload. Staff skills and competencies can also be recorded and tracked.

Figure 2.2-4 Staff Management
The Service Management module supports core case management processes. The service management module includes protective services reporting, intake, and investigation, as well as intake processing for other service streams or voluntary self-referrals. The service management module also provides functionality supporting the delivery of ongoing services to children and their families by staff or licensed providers.
The Provider Management component supports the recruitment, licensing and contracting of providers. The provider management module also assists workers in making placement decisions or service arrangements with child welfare and family service providers.
The Financial Management functions support the business of child welfare and family services, by allowing users to perform fiscal management tasks, assure proper payment for services, and claim associated federal reimbursement. Building on the relationships that exist among the services provided to a participant, the providers of the service, the terms of provider agreements, and the funding source of the payments, the design captures information that can result in the generation of payments to providers for ongoing services.
In addition to functional and processing components, the design supports interface components that exchange information with other automated systems.
2.3 FACTS Intended Use and Operation

FACTS will offer consistency throughout all functions and components which will assist Workers and Supervisors in tracking individual children and families from the point of initial contact to the point of case closure. These features, although different in service delivery options, are similar in nature over the life of a case. They include a “service plan” driven system, views that show relationships among cases, clients and families, search and retrieval functions that meet the needs of workers and supervisors, the replacement, standardization, and reduction of forms, automation of the CYFD Family Risk Assessment, Case Plan and Safety Plan, automation of case file folders, and automation of the following processes: licensing, Title IV-E Foster Care Maintenance Determination/Redetermination, and fiscal and accounting functions including eligibility for services and fiscal reporting.

When implemented, FACTS will be functionally robust and responsive to State needs for a system to support case management. The application will support automation of casework practice, and integrates client, service, financial, and provider data. An easy-to-understand, easy-to-navigate graphical user interface will guide users through the basic functions of casework, such as investigation, planning, and placement. The design will allow users to record data necessary for detailed client tracking and monitoring, enhanced resource planning and management, and state and federal reporting. In addition, the use of FACTS will facilitate effective practice delivery for the children of CYFD.

2.4 Planned FACTS Deployments

The system will be deployed in a statewide uniform manner across the 57 field offices, 30 county offices, in each of the CYFD 5 regions and State Headquarters. It will also be used by the providers, the courts and other interested stakeholders.

2.5 Outsourcing

Once FACTS is deployed, maintenance, enhancements and support of the application can be performed in one of two mechanisms. The first option is to utilize the traditional roles of DoIT for Infrastructure support and ITS for application maintenance. The increased staffing needed to support the new equipment and maintenance procedures may preclude the use of these organizations. In which case, ITS could consider outsourcing some or all of these functions. A vendor similar to an Application Service Provider (ASP) could perform these functions. These functions are not normally considered ASP provided services. Generally, these vendors offer “turnkey” solutions involving a specifically marketed application product such as SAP or PeopleSoft so the ability of a particular vendor to meet ITS requirements must be assessed.
Some of the additional criteria that should be considered in evaluating a potential outsourcing ASP are:

- Total cost of service
- Quality of service delivery as determined by interviewing others who have used the contractor,
- Industry knowledge i.e., child welfare,
- Technical knowledge i.e., SACWIS technology architecture, web services. SOA
- Project management skills,
- Skills of personnel who will be assigned to the project,
- Financial health of provider,
- Quantity and quality of current State Information Technology staff, and
- Service Level Agreements.

ITS will need to determine whether “application hosting” will be sufficient to support FACTS. It may be necessary to contract out maintenance services. If so, combining these efforts could provide significant cost savings. These options will need to be determined at the initiation of implementation.
3 Critical Success Factors, Risks, System Design Decisions, Assumptions and Constraints

The SAS lists the factors critical to the success of the architecture development process. These factors include assumptions, dependencies and constraints identified, and risks assessed, along with recommendations to ensure that success criteria are achieved, and identified risks are mitigated.

This document will assist ITS and potential future vendors to understand what was included in the decision and to be aware of important factors to maintain system integrity as FACTS is modified and extended. It will also help them know what was NOT included in the decision, so that if new factors become evident there will be no artificial barriers to changes.

3.1 FACTS Architecture Development Critical Success Factors

The following delineates some of the key factors that have been identified to ensure a successful implementation of the FACTS system architecture:

- Iterative Approach: The components of a system architecture are highly interdependent, so creating a system architecture is highly iterative. For example, the desired functionality affects needed software, database and network components. Additionally, hardware decisions influence the software architecture. Iterative application development is designed to provide results quickly, handle risks early in the project, facilitate development in ‘Internet time’, and allow for growth without significant impact to existing functionality or performance.
- Communications among State Departments and all vendor personnel to support the technology transition that will take place as part of the implementation of the system,
- The ability to scale as volumes grow or shrink and handle increases or decreases from the data source to the desktop with little effect on initially accepted response times,
- The ability to interface to and share data with existing Legacy Systems during system rollout and conversion of data from the Legacy System.
- The ability to coexist with other State agency applications residing on the desktop without creating configuration issues or response time issues,
- The ability for offices and agencies within the FACTS scope to operate continuously in the event of a system interruption within another office,
- Satisfactory response time, and
- Flexibility to help handle the constant change occurring in both the industry and the State.
3.2 FACTS System Design Decisions, Assumptions & Constraints

3.2.1 Decisions
Take advantage of existing CYFD and State network infrastructure and resources, leveraging the significant investment the State already has in place.

3.2.2 Assumptions
Scope and Detail of SAS – The SAS will identify software to the level of component, not to class or object level. Components are defined in the FACTS Application Architecture as service objects supporting a business function. See Figure 2.2-1 for some examples of business level components, i.e., Intake, Investigations. Definition of classes, their interfaces, message signatures, and operations, as well as physical database design, will be part of Conceptual and Detail Design.

Network Management – CYFD/ITS uses its own underlying telecommunication network. At this time, network monitoring is not performed as part of a proactive approach to identify and resolve problems as they are detected. Instead, a reactive procedure involving calls made to help desk personnel refer problems to the telecommunications group who then deciphers whether a problem exists. This can slow the response to resolve a situation that could be resolved remotely and could have been identified before the user community realizes a problem even existed. Detecting these issues in advance of processing issues will become a more critical component of operating a web-based application. Knowing a problem exists before the application executes a process, dependent on networked equipment availability, will save many hours of error correction and re-processing. It is recommended that CYFD invests in a monitoring capability to detect network issues as they occur.

3.2.3 Constraints
The primary constraint is the funding that CYFD can acquire to complete the implementation of the FACTS project. A secondary constraint is the continuing State support not only to complete development and implementation but to operate and enhance FACTS, if needed. Potential staff reductions and budget constraints would make it difficult for CYFD to support the FACTS application.

3.3 Risks
Staff Availability: Due to budgetary constraints, staff constraints have been instituted and are expected to continue. These restrictions seriously reduce the likelihood of successful implementation because of the need for full time staff to assist fellow employees in the transition to the system. Also, the availability of CYFD staff with adequate skill set and experience level is a significant risk.

Security: The selected architecture makes available application capabilities that were not previously possible. In order to implement these capabilities, many new features will need to be put in place to ensure the security of the entire enterprise. Fortunately, the mechanisms all exist to protect the enterprise but they require additional software to
implement and expertise to manage. Based on the prior risk and the inherent problems with unfamiliar, new security practices, there is potentially a significant risk in this area.

**Maturity:** Software products have long been measured by their longevity in the marketplace. This has been evolving recently because the software marketplace has been changing so rapidly to keep pace with improved technologies. This does not mean that maturity is not desirable, but in today’s market, it is very difficult to find where new technology is concerned. It would be remiss not to identify maturity as a risk but in considering the speed that technology is changing maturity will not hold back those organizations needing to implement that technology.

**Application Response Time:** Expectations for system performance (transaction response times) as described in the APD may be unrealistic since there are so many factors beyond the control of CYFD’s Information Technology Services to manage. The network is subject to impact from the activity of other applications using the network. This makes the management of the network somewhat unpredictable compared to systems that operate entirely in a homogenous environment or as a single client whether standalone or connected strictly to a LAN.

Without proper design and planning, it is unlikely that any internet application will meet aggressive response time requirements. Of course, n-tier applications are valuable for reasons other than instantaneous response. Their reliability, scalability and availability are unmatched because the n-tier architecture allows for upgrading, clustering and migrating as application needs for capacity or security change.

Users, however, require a responsive application and will often revolt against one that is not responsive. The minimum achievable response time is expected to be three seconds. To achieve this response time, the application design will need to consider keeping the data transmitted to a minimum with each application function. To ensure performance, application processes will be reviewed during detailed design to determine if the design can be improved to reduce the amount of data to be transmitted in a single transaction.
4 Business Architecture

4.1 Business Architecture Description

The Business Architecture describes, at a conceptual level, the functions to be performed on data by the planned system, and its interactions with users. It may break the system down into smaller pieces (packages or subsystems) and show how they interact. Functionality is described using design models such as system context diagrams, use cases, class diagrams, object models, user interface models, functional decomposition, or models that show information flow. Through these techniques, created during Conceptual and Detailed Design phases, system scope is established, and scope and functionality to be supported by the system are communicated to stakeholders and validated against original business objectives. Please refer to the current FACTS documentation for details on the approach and notation for modeling application functionality. Representative documentation can be found as follows:

- **Current Topic Papers:**
  G:\ADS\RELEASES\Release v3700\Topic Papers\Completed

  The topic papers are updated during each release and once the Release 4.0.00 updates have been completed; those topic papers will be located in:

  G:\ADS\RELEASES\Release v4000\Topic Papers\Complete

  This occurs for each release.

- **Release 4.0.00 Requirements Documents:**

  The requirements documents for Change Requests included in Release 4.0.00 are located in:

  G:\ADS\RELEASES\Release v4000\Requirements

  The requirements documents are organized by change request and program area. There are folders for each CR and the requirements document is the document that is named CRXXXReqntx.xls in each CR folder.

  Requirements documents can be accessed for prior releases in the related release requirements folder.

- **Release 4.0.00 System Design Documents:**

  The system design documents for Change Requests included in Release 4.0.00 are located in:
The system design documents are also organized by Program area/Change Request. In each CR folder, the system design document is named "R4000CRxxx-Description.doc"
Example:
G:\ADS\RELEASES\Release v4000\System Design\Change Request\FSD\CR1064 FSD JJS Payments\CR1064 System Design v4.0.doc

System design documents can be referenced for any release in the release related folder.

- **Batch Design Documents:**

  G:\ADS\Batch\FACTSBAT\ProdDocs

  These documents are stored by program name

- **ER Diagrams:**

  G:\ADS\ERDs

  This directory contains Visio generated entity relationship diagrams. They are not current and will not be updated until FACTS.

Organizational flexibility is enhanced when software applications are independent of the technical components. Ideally, there should be little effect on an application if a technical component, such as the TCP/IP stack, is changed or updated. Therefore, while the Hardware and Network, Software and Data Architectures must allow the application to implement its full functionality, the need for speed and flexibility require as little coupling as possible between the architectures and the application.

*Since the Business Architecture is created as part of the application development cycle, and the application will ideally be independent of the underlying frameworks, the Business Architecture detail is not included in the SAS.*

### 4.2 Relationship of Business Architecture to Other Views

The ability to share common components (network, hardware, software and services) among several applications is a key to developing high quality, fully functional applications quickly. The Hardware and Network Architecture and Software Architecture must be based on substantially open systems components (i.e., components that can be exchanged with other components that provide equal or similar functionality, with little or no negative impact to the application).
Following are the descriptions of the relationships between the Business Architecture and each of the other architecture components, and how these relationships support meeting functional requirements.

4.2.1 Business Architecture and Application Architecture

The high-level functional characteristics of the application provide information on how to select the appropriate Application Architecture. In turn, the Application Architecture helps to identify the technical components, in three critical areas, that will help the application deliver its functionality:

- The networks, hardware and software required to deliver functionality for each of the high level ‘tiers’ of the application (“Application Partitioning”),
- The software and organization to protect and provide access to the system users (the Security Model), and
- The software products needed to communicate with other applications or platforms, including legacy applications (“Interface Architecture”).

The relationship between the Business Architecture and each of these critical areas is described below.

4.2.1.1 Business Architecture and Application Partitioning

The Business Architecture (or high-level system requirements) provides input into the Application Partitioning decisions. For example, FACTS is an n-tier application with over 2000 user PC’s; it will grow over time; it must be flexible and able to quickly respond to changes in State or Federal programs; and it must be able to incorporate new technologies. These are functional characteristics that suggest partitioning the application into multiple tiers on open systems with an ultra-thin topology (i.e., a web-browser-based user interface).

At a high level, the Application Architecture provides guidance to the application designers and developers: what tools should be used for development, what platforms for deployment, and what code segments can be best packaged together. With many physically separate computational nodes and like functionality split across several environments, an analysis should be done in application build time during Phase 2 to specify the detailed packaging of functionality by node. If such an analysis proves necessary for any part of FACTS, the results of this analysis should be represented by a UML Deployment Diagram.

4.2.1.2 Business Architecture and the Security Model

The functional characteristics provide input into the Security Model, which describes the service available to the application to protect system assets; i.e., by restricting options in the firewall, user interface design, controlling access to data, or encrypting data for transmission. The Security Model also describes the administrative work necessary to
organize users according to responsibilities or access rights, and may have impact on
the tools chosen to build and deploy the application.

4.2.1.3 Business Architecture and Interface Architecture

The architecture must allow FACTS to exchange data with many different agencies and
systems, using the technology now in place to implement such interchanges, and also
allowing for enhancements to these interfaces as well as new types of interfaces in the
future. The Business Architecture specifies these interface strategies. The FACTS
Overall Context Diagram depicted in Section 5.4.2 Application Architecture summarizes
FACTS interfaces; the related tables show which interfaces require what technologies.
The technologies required to implement the interfaces are specified in the Hardware
and Network Architectures and Software Architectures (see Section 6 and 7
respectively).

4.2.2 Business Architecture and Hardware, Network and Software
Architecture

The hardware, network and software architectures provide the platform on which the
application runs. The application software should be independent of the hardware on
which it executes so that the application can be easily transported to a new platform if
terprise requirements dictate. The Software Architecture should consist of open
system components with standard interfaces. It is required that any component of the
network, hardware or software architecture can be substituted with another component
that provides the same or similar functionality. For example, an existing processing
platform can be ported from one product to another without affecting the application
environment.

Application partitioning is one view of the Application Architecture, overlaid on the
Hardware and Network and Software Architectures. The Environment Models provide
another view that includes the application's functional software (at the logical tier level)
and platforms. If application functionality is distributed in complex fashion across
multiple nodes, additional UML models may be required.

To ensure that the FACTS architecture will include the necessary technology to provide
all required functionality, FACTS requirements were analyzed for technical implications.
As described in section 1.6.2.1 Review Technical Requirements, this process is not
scientific, but relies on the experience and knowledge of the analysts. The analysts
have already identified those requirements that may have significant technology
implications and are documented in (Appendix 12.24: Technical Requirements for
FACTS.)

Architecture Validation meetings will be held with State stakeholders. In these
meetings, technologies that will be required for FACTS to provide functionality will be
identified.
4.2.3 Business Architecture and Data Architecture

Data can be thought of as the “what” in data processing, and the Business Architecture as defining how the data is to be processed. Functionality is often documented by describing the data required to complete the process, how the data is acted upon, and the final state of the data. In the FACTS project, this information is captured in the Conceptual Design through use cases, work flows and interaction diagrams. These artifacts provide the main source of input for the logical data model, which consists of the entities to be acted upon, their attributes, the relationships between them and their cardinality within each relationship. Detailed Design of the Business Architecture allows development of the physical data model and definition of the tables. This does not currently exist, but will be done for FACTS.

As stated above, the business and data architecture should be decoupled so system architecture changes and functional changes do not have significant impact upon each other. This methodology can be followed for all architecture areas except security. Security requires specific building blocks be defined for the technical and software architectures ensuring their protection throughout the entire FACTS system. The six building blocks and their incorporation into the technical architecture are documented in Section 5.5.7 and the FACTS Security Plan.
5 FACTS Application Architecture

5.1 Application Architecture Description

The application architecture describes the application in terms of functional partitioning, interface processing and security processing. An overview of the three application architecture components follows:

- Functional partitioning provides guidance to the designers and builders on how to package and deploy application functionality. The Application Architecture identifies the tiers in the application topology, and assigns the work done by the tiers to platforms. This analysis ultimately determines the physical location where software will execute.
- The Interface Architecture ensures that FACTS can perform all necessary interface functions. Interfaces are identified and categorized by type and technology, and a strategy for handling each type of interface is devised. Interfaces can be across platforms within the application, or with other applications. Other applications may belong to ITS, other State agencies, non-governmental entities, other states, or the federal government.
- The Security Architecture addresses the six security building blocks:
  - Authentication - Determining whether individuals are who they say they are,
  - Authorization - Determining what access privileges have been granted to a user, group or process and enforcing these privileges,
  - Confidentiality - Protecting data from unauthorized eavesdropping,
  - Integrity - Assurance that data is not modified (by unauthorized persons) during storage or transmittal,
  - Auditing - Creating a chronological record of all the activities of a system, and
  - Non-repudiation - Proof of participation in an electronic exchange of information

What was the overall approach that was used in defining the Application Architecture? How were the components defined? What were the decisions and assumptions that were made and the constraints that were imposed in defining/selecting the application architecture? Figure 5.1, below, is a graphic representation of the Architecture Selection Methodology that was used for FACTS (and will be used for FACTS) to develop the Application Architecture. The following sections in this chapter will discuss these topics.
5.2 Application Architecture Approach

Technical meetings with State stakeholders were conducted to ascertain the direction that the architectural design team should take in specifying a final design. The first decision point was to hone in on the architecture type (i.e. fat client, plump client, thin client, ultra-thin client). See the Glossary in Appendix 12.1 for definitions of these architecture types.

The thin and ultra-thin client models were selected as the application architecture partition. Following the selection, a number of additional issues were addressed as part of the selection process. These issues were brought out during the selection process and needed to have a resolution or identification of when they could be resolved. See Section 5.6: Application Architecture Decisions documenting the entire selection process.
5.3 Partitioning of Design Elements

FACTS was deployed as a client/server application. See Figure 5.3.1

Figure 5.3.1 Current Topology

FACTS will be deployed primarily as an n-tier application with an ultra-thin topology, except where functional requirements cannot be met with this topology. In that case, the application should have a small footprint on the user interface device.

Among the various architectures, the n-tiered architecture is the generally accepted model. Using this approach, an application can be split into layers for presentation,
business logic and database management. Each layer is independent and has defined interfaces for communication. This architecture facilitates changes in either the presentation, business logic or data access layers without affecting the other areas. Thus, new presentation services, via Web browser, telephone interfaces, etc., can be plugged-in to the application without affecting the business logic and database functions. Similarly, new database servers can also be added without affecting the presentation and business logic layers. This type of architecture enhances scalability, openness and flexibility.

All functionality provided by application software can be viewed as falling into one of five classes:

- **Presentation Management** - Presentation is the human-machine interface.
- **Business Rule Enforcement** - The business rules are the statements of policy which govern:
  - Enforcement of contextual data validation not addressable via data item characteristics and data structure definitions found in the data dictionary,
  - Business Rules that support the determination and levels of eligibility, and
  - Determination of the software component to which control will be passed under specific conditions.
- **Data Validation** - Data validation functions ensure that data values, input through a presentation medium, are appropriate prior to their use in business rule enforcement or storage/deposition in shared databases.
- **Data Access** - Data access focuses on retrieving data from the data structure.
- **Process Control** - Process control is unique in this list, in that it not only is a structural element within a program; it is the glue which governs the interaction among all other structural elements of an application. In the world of FACTS, service compositability and process orchestration which be the ‘structural glue’ that will hold the business process together.

These classes of functions or elements of design as they are referred to during the software design process, serve as the initial criteria for partitioning of software during design and subsequent construction. Partitioning of software by function in this model may result in a single software component to provide functionality in all classes or a number of software components for each class.

Software developed to implement ITS business functions must:

- Be partitioned by class of functionality,
- Be able to execute under multiple tiers and be portable to each tier, if applicable,
- Propagate changes to other tiers through automated tools without the need for re-coding through manual assistance or intervention, and
- Have a small application footprint on the desktop device.

It should be noted that FACTS will be a web-enabled application based on a SOA foundation. Applications, therefore, will be implemented via business services.
Partitioning of the five design elements is illustrated in Figure 5.3-2: Application Architecture Partitioning. The building blocks represent design elements, with logical tiers represented by the braces on top and physical platforms represented by the braces on the bottom. Figure 5.3-2 illustrates:

- Presentation Management elements are part of the logical Presentation tier and are handled by the web browser on the desktop,
- Data Validation Rules elements belong partly in the Presentation logical tier and partly in the Business Logic logical tier and are handled by the web browser and the web server,
- Business Rule Enforcement is part of the Business Logic logical tier and is handled by the application server and partially by the web server,
- Data Access elements are part of the logical Data tier, and are partially executed on the application server, and partly on the database server, and
- Control elements, the glue that holds the other elements of the application together, exist in all logical tiers and on all platforms, except the data layer and database server.

Further discussion of these design elements follows:

### 5.3.1 Presentation Management

Presentation Management is responsible for the immediate user/computer interface. In the ultra-thin topology, the web browser handles this interface. FACTS is to be developed in such a way that it is not dependent on any particular browser or its unique functionality. This is not to imply older releases of browsers should be supported. In fact, the latest releases available should be used because of the increases in capability they normally provide. However, current releases often have functions not compatible within the other browser and therefore that capability should not be utilized to develop FACTS. The following presents requirements for the ADE/Presentation Management module:

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controls:</td>
<td></td>
</tr>
<tr>
<td>• Menus</td>
<td></td>
</tr>
<tr>
<td>• Tree Views</td>
<td></td>
</tr>
<tr>
<td>• Text Box - Single and Multi-line</td>
<td>To support the broadest interaction with the user.</td>
</tr>
<tr>
<td>• Labels</td>
<td></td>
</tr>
<tr>
<td>• Drop Downs</td>
<td></td>
</tr>
<tr>
<td>• List Boxes</td>
<td></td>
</tr>
<tr>
<td>• Radio Buttons and Check Boxes</td>
<td></td>
</tr>
<tr>
<td>• Tab Folders</td>
<td></td>
</tr>
<tr>
<td>• Frames</td>
<td></td>
</tr>
<tr>
<td>Capabilities Required</td>
<td>Comment</td>
</tr>
<tr>
<td>---------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Wizards</td>
<td>To improve developer productivity and standardization.</td>
</tr>
<tr>
<td>Toolbars</td>
<td>To emphasize commonly used application functions.</td>
</tr>
<tr>
<td>Property Dialogs</td>
<td>To list available properties for use by the developer and improve productivity.</td>
</tr>
<tr>
<td>Object Explorer</td>
<td>To list and manipulate the Object Hierarchy.</td>
</tr>
<tr>
<td>Support ECMA</td>
<td>To support a wider market of developers.</td>
</tr>
<tr>
<td>Support Drag and Drop within the browser</td>
<td>To improve productivity of users.</td>
</tr>
<tr>
<td>Provide DHTML/HTML Generator Tool</td>
<td>To improve developer productivity.</td>
</tr>
<tr>
<td>Support Multiple Fonts</td>
<td>To support the broadest interaction with the user.</td>
</tr>
<tr>
<td>Context Sensitive Help</td>
<td>To assist users in navigating and understand how to complete a business function.</td>
</tr>
<tr>
<td>Support for Wireless Access</td>
<td>To expand the access capabilities to mobile users.</td>
</tr>
</tbody>
</table>
5.3.2 Data Validation Rules

In the ultra-thin topology, Data Validation Rules are implemented partially in the presentation software (i.e., the web browser) and partially in the web server. On the web browser, for example, it’s fairly simple to ensure that a phone number only includes the right number of numerals and the correct formatting characters (dashes or parentheses) in the correct locations. A good principle to follow is to validate data as close to its source as the application development tool or execution environment will allow. Using the presentation layer for basic data validation can provide better performance and more intuitive behavior. This is not to be construed to mean that customization of the Presentation Management software is an acceptable solution.

Data Validation Rules on the web server ensure that the basic format of data input through the user interface is appropriate for further formatting. To use the simple phone number example, Data Validation Rules could ensure that the area code is valid, and the exchange valid for the area code is entered.

5.3.3 Business Rule Enforcement

Business Rules implement the data validation and business policies of the enterprise; they go beyond the simple structural validation of Presentation Management to ensure that data values are valid within the context of the application. In the simple example of the phone number input by the user, Business Rule Enforcement may ensure that the area code falls within a specific geographic area. Business Rule Enforcement also dictates the flow of control within the application; i.e., the specific processing to be performed against data, based upon its characteristics or value.
Some Business Rules may be implemented on the web server, but most will be implemented on an application server. The following presents requirements for the ADE/Business Logic module:

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Management</td>
<td>To improve developer productivity.</td>
</tr>
<tr>
<td>• Date Function</td>
<td></td>
</tr>
<tr>
<td>• String Functions</td>
<td></td>
</tr>
<tr>
<td>• Mathematical Functions</td>
<td></td>
</tr>
<tr>
<td>• Type Conversions</td>
<td></td>
</tr>
<tr>
<td>Support Object Modeling Interface to Rational Rose</td>
<td>To standardize development of object classes and quickly build foundation of application business logic.</td>
</tr>
<tr>
<td>XML Support and Generation</td>
<td>To improve standardization of data and processing of that data.</td>
</tr>
<tr>
<td>SOAP - Simple Object Access Protocol</td>
<td>To provide a framework for developers to define XML messages and execute processing on the data contained in the XML.</td>
</tr>
<tr>
<td>CSS and XSL Support and Generation</td>
<td>To provide developers with standard mechanisms to present, validate and process data.</td>
</tr>
<tr>
<td>Thread/Serialization Management</td>
<td>To improve control of this extremely process intensive coding technique.</td>
</tr>
<tr>
<td>Built-in Self Documentation Support</td>
<td>To improve maintainability.</td>
</tr>
<tr>
<td>Support Error Handling Construct</td>
<td>To improve stability and maintainability of the application.</td>
</tr>
<tr>
<td>• Try</td>
<td></td>
</tr>
<tr>
<td>• Catch</td>
<td></td>
</tr>
<tr>
<td>• Finally</td>
<td></td>
</tr>
<tr>
<td>Heavily Typed Language Support</td>
<td>To improve application stability.</td>
</tr>
<tr>
<td>Capabilities Required</td>
<td>Comment</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Support Object Oriented Development Principals</td>
<td>To support UML design process.</td>
</tr>
<tr>
<td>• Encapsulation</td>
<td></td>
</tr>
<tr>
<td>• Inheritance</td>
<td></td>
</tr>
<tr>
<td>• Polymorphism</td>
<td></td>
</tr>
<tr>
<td>Built-in Memory Management and Garbage Collection Support</td>
<td>To improve application stability.</td>
</tr>
<tr>
<td>Line by Line Debug Mode for:</td>
<td>To improve developer productivity.</td>
</tr>
<tr>
<td>• Objects</td>
<td></td>
</tr>
<tr>
<td>• Server Pages</td>
<td></td>
</tr>
<tr>
<td>Developer Tool</td>
<td>To improve developer productivity by generating code to define classes, attributes and methods as well as integrating server page development application server processes.</td>
</tr>
<tr>
<td>Impact Analysis Tool</td>
<td>To improve developer productivity by identifying the impact of code changes across the entire development project from listings of effected classes, methods and attributes.</td>
</tr>
<tr>
<td>Scalability of Application hardware to address increased user loads</td>
<td>To improve performance.</td>
</tr>
<tr>
<td>Portability of Application code</td>
<td>To improve manageability and maintainability.</td>
</tr>
<tr>
<td>Fault Tolerance and Fail Over of Web and Application Servers</td>
<td>To manage/improve availability/reliability.</td>
</tr>
<tr>
<td>Load Balancing of Web and Application Servers</td>
<td>To improve performance.</td>
</tr>
</tbody>
</table>

**5.3.4 Data Access**

Data Access design elements address the need for the application to store and access data from shared or non-volatile (persistent) storage locations (as opposed to transient, or session, data). Some Data Access functions may be implemented using stored procedures that reside in the database itself. Stored procedures may be simpler to write, provide more efficient operation, or provide functionality that may be difficult to implement in other ways.
Access across platforms requires the implementation of middleware to support three types of communications:

- **Synchronous**, which allows for 'conversational' access. At a minimum, this implies the need to send a request to a server, and a response back to the requestor or client while the requestor waits.
- **Asynchronous**, which in its most simple form is a unidirectional message. This approach is ideal for maintaining consistency between data sources during a period of conversion, or to enable event-driven processing.
- **Batch**, which supports periodic processing where a user’s input is not required to execute the process
The following presents requirements for the ADE/Data Access module:

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support for Data Types • Character • String • Integer • Long Integer • Single and Double Precision Decimals • Blob</td>
<td>To ensure all Application data can be stored.</td>
</tr>
<tr>
<td>SQL Generator</td>
<td>To improve developer productivity.</td>
</tr>
<tr>
<td>Connection Pooling</td>
<td>To improve performance.</td>
</tr>
<tr>
<td>Query Analyzer</td>
<td>To improve performance.</td>
</tr>
<tr>
<td>Database Engine Monitor</td>
<td>To manage application resource utilization.</td>
</tr>
<tr>
<td>Backup/Restore Processing</td>
<td>To protect application data.</td>
</tr>
<tr>
<td>Publish/Subscribe Replication Method</td>
<td>To support decentralized application.</td>
</tr>
<tr>
<td>Offline Database Re-synchronization</td>
<td>To support offline traveling users.</td>
</tr>
<tr>
<td>OLAP Tools</td>
<td>To support decision support data analysis.</td>
</tr>
<tr>
<td>XML Support</td>
<td>To reduce/eliminate data transformation processing within the application.</td>
</tr>
<tr>
<td>Row Level Locking</td>
<td>To control data access most effectively for users.</td>
</tr>
<tr>
<td>English-Like Query Language Support</td>
<td>To improve usability.</td>
</tr>
<tr>
<td>Report Generator/Writer</td>
<td>To improve data access capabilities.</td>
</tr>
</tbody>
</table>
### Capabilities Required

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indexed Views</td>
<td>To improve performance.</td>
</tr>
<tr>
<td>Soundex Search Keys</td>
<td>To improve search results.</td>
</tr>
<tr>
<td>Fuzzy Search Logic (i.e. Search Bob and find):</td>
<td>To improve search results.</td>
</tr>
<tr>
<td>• Bob</td>
<td></td>
</tr>
<tr>
<td>• Bobby</td>
<td></td>
</tr>
<tr>
<td>• Robby</td>
<td></td>
</tr>
<tr>
<td>• Rob</td>
<td></td>
</tr>
<tr>
<td>• Robert</td>
<td></td>
</tr>
<tr>
<td>ERD/Object Modeling Integration</td>
<td>To synchronize logical, physical and object models.</td>
</tr>
</tbody>
</table>

### 5.3.5 Process Control

Process control elements do not provide business functionality; rather they provide the programmatic framework under which the business functionality is delivered, and serve as the glue connecting the structural elements of the application. Examples of Process Control elements include resource management (e.g. connection pooling or memory management), common services routines (e.g., formatting printed output, converting XML to HTML, processing an initialization or configuration file), routines for controlling sequencing or routing within the application, and services for inter-process or inter-platform communications.

Application servers provide some Process Control functionality, such as concurrence management (managing multiple processes and threads (sub-processes) within the application), resource pooling (e.g., memory, data base connections, or connections to queuing software), startup and shutdown logic, and connectors to databases and platforms.

Process Control elements may be required in any logical tier or any platform, although generally they are not necessary on the Database Server; if required, they would generally be implemented as stored procedures that include some procedural language (other than SQL) or user-defined functions.

The following presents requirements for the ADE/Message Oriented Middleware module: See Appendix 12.17, Middleware Questions for a list of questions that maybe submitted to the vendors.
<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Integration Support</td>
<td>To support data access from 'blade server'-based legacy systems.</td>
</tr>
<tr>
<td>Guaranteed Message Delivery</td>
<td>To enable “submit and forget” processing to improve application performance.</td>
</tr>
<tr>
<td>Monitoring Tools</td>
<td>To improve manageability.</td>
</tr>
<tr>
<td>FTP</td>
<td>To enable send and receiving data files.</td>
</tr>
<tr>
<td>EDI</td>
<td>To describe data files to sending and receiving parties.</td>
</tr>
<tr>
<td>Internet</td>
<td>To provide a mechanism to transmit data outside the State network.</td>
</tr>
<tr>
<td>Digital Media</td>
<td>To temporarily store intermediate data files.</td>
</tr>
</tbody>
</table>

### 5.4 Interfaces

FACTS will have interfaces with the federal government, other state agencies, other applications within ITS, and within FACTS and other non-government entities. An interface is an exchange of data where both parties agree on the exchange medium (e.g., cartridge tape or electronic interchange), protocols (e.g., HTTP, WAP, TCP/IP, SOAP), record layouts, timing, and frequency. The purpose of the Interface Architecture is to ensure that FACTS has the technology available to successfully complete all the data exchanges required when implemented, and be flexible enough to enhance existing interfaces or easily add additional interfaces using new or different technologies. See Figure 5.4.3-1, Interface Details.

#### 5.4.1 Interface Architecture Approach

The desired FACTS interfaces are documented below.

The CYFD SACWIS will contain a common GUI interface to provide users with a mechanism for entering a variety of related applications. The SACWIS Interim Final Rule requires that SACWIS provide for timely and automated exchange of information between SACWIS and the following systems: Title IV-A (TANF), Title IV-D (child support), Title XIX (Medicaid) and NCANDS. The Medicaid information system is implemented on a different system under another agency: HSD. These systems can, however, be requested to accept periodic exchange of data between systems using interfaces which translate data from the format and context of one system to the appropriate format and context of the target system. All systems must maintain reference data for the other system in order to enhance interface efficiency, for example the Title IV-A Member ID or the Title IV-D Case ID.
Interface requirements are typically addressed in the following ways:

- **Batch:** Periodic (i.e. daily, monthly, annually) transfer via tape or electronic batch file between SACWIS and (IV-A), (IV-D), Medicaid (Title XIX), etc. These interfaces involve file transfer between systems. These data transfers provide the vehicle for the exchange of information among the systems. Each system determines whether they consider the interface data to be the best source of information and updates the target system, or discards the interface information.

- **Near real-time:** User requested electronic transfer of information between SACWIS and (IV-A), (IV-D), Medicaid (Title XIX), etc.

These transactions can consist of the following:

- **Synchronous:** Real Time Synchronous (RTS) transactions provide a mechanism to constantly interface two or more systems. Therefore, transactions are completed on demand (whenever a system has data to send).

- **Asynchronous:** Real Time Asynchronous (RTA) transactions provide a mechanism to schedule interactions, based on a time algorithm (for example: every so many minutes), between systems. This capability enables extremely flexible system interfacing.

To meet the interface requirements of Batch and near real-time transactions (RTS/RTA), Message-Oriented Middleware (MOM) is recommended. MOM provides reliability and flexibility through the separation of systems and data unspecific messages. For example, the operations of System A is not dependent on System B, which System A is trying to interface with. Therefore, System A continues with normal operations, leaving MOM to continue with the transaction process even if System B is not ready to be interfaced. MOM can also package any type of data (i.e. text document, images, etc.) in a “message” for a transaction. As the complexity of the communications environment escalates, an enterprise service bus (ESB) will be employed.
5.4.2 FACTS Overall Context Diagram

A context diagram shows the “business problem domain” and the external agents that interact with it. It is the very highest level of data flow information. The notation used is a hybrid of DeMarco/Gane & Sarson, as described in the following table.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow</td>
<td>Describes the flow of data; i.e. inputs to/outputs from FACTS.</td>
</tr>
<tr>
<td>Octagon</td>
<td>Similar to the Actors defined in the Unified Modeling Language (UML): individual users, agencies, organizations or systems that interact with FACTS.</td>
</tr>
<tr>
<td>Large Rectangle</td>
<td>Specifies the FACTS system. Consult SRD Business Architecture.</td>
</tr>
</tbody>
</table>

Please see Appendix 12.1, Glossary for a description of the acronyms used in the Context Diagrams (Figure 5.4.2-1).
**Batch Interfaces**

Figure 5.4.2-1: Batch Context Diagram

*Child Support and Enforcement Division*
5.4.3 Interface Details

The following table provides a suggested approach in which FACTS may interface with other systems. The table shows possible interfaces and their requirements (Batch, RTS, RTA), points of interface, possible exchange mediums and their protocol, frequency of transactions, and volume. The interface and points of interface information, as they currently exist, have been provided through knowledgeable State Staff. The other information depicted are semi-controllable variables, where the values are seen as the “best fit” with current systems’ architecture. As the Design Phase proceeds, more information will be gathered to verify the table and its data. This document will be updated when the information can be obtained.

<table>
<thead>
<tr>
<th>Interface Name</th>
<th>Batch (B)</th>
<th>Point of Interface</th>
<th>Exchange Medium</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title IV-D</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
<tr>
<td>AFCARS</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>Direct Connect</td>
</tr>
<tr>
<td>NCANDS</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
<tr>
<td>ACF801</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>Direct Connect</td>
</tr>
<tr>
<td>Child Care</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
<tr>
<td>MMIS</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
<tr>
<td>APS Billing</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
<tr>
<td>CPS Billing</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
<tr>
<td>Title XIX</td>
<td>B</td>
<td>Microfocus COBOL</td>
<td>Datasets</td>
<td>FTP</td>
</tr>
</tbody>
</table>

Figure 5.4.3-1: Interface Details

5.5 FACTS Security Model

Security is not just about throwing technology at a problem (such as adding Public Key Infrastructure (PKI), firewalls and Secure Sockets Layer (SSL) to make a system more secure). Because of the sensitivity of the data contained in the system, a set of six, core interrelated ‘security building blocks’ are needed for secure transactions from end to end. See Figure 5.5-1, The Security Building Blocks. These security building blocks consist of the following, and are discussed in detail in the sections that follow:
5.5.1 Authentication

The identification/authentication security building block comes into play at the beginning of a secure transaction and is arguably the most important security building block. Identification is the process of presenting the identity of an individual in a transaction. An example of a specific identification process is a server that identifies users by utilizing unique machine-readable user names. The problem with using identification alone is that other users and/or systems do not have assurance that participants are
legitimate users and/or systems. To solve this problem, the authentication process comes into play. Authentication is the process of verifying the identity of various participants (users/systems) exchanging information. An example of authentication is when a computer network user enters a password for authentication along with a user name for identification.

5.5.2 Authorization
Authorization is the second building block of a robust security system. Once a security system has validated a legitimate user to the network, the next step is for the security system to verify what actions the user is able to perform and what information the user is able to access. Access privileges are usually based on a user’s role in the organization via a previously defined access control system. The access control system determines a valid user’s access privileges, then makes sure that the user only performs actions and sees the data he or she is allowed to see. Once an authorized user has initiated a transaction, the user will need another security building block to ensure that the contents of the transaction have not changed. This security building block is called integrity.

5.5.3 Integrity
As a party to a transaction, some level of assurance needs to be provided that the contents of the transaction have not been altered in some way by an unauthorized user. The integrity security building block provides assurance of the correctness of the content and/or source of a piece of information. Hash values are the primary way people ensure data integrity. Another security building block, closely related to integrity, is confidentiality.

5.5.4 Confidentiality
Confidentiality, or privacy as it is sometimes called, is the process of keeping information secret from all but those who are authorized. It is the fourth building block to a secure transaction. A common example of securing electronic information is cryptography.

Confidentiality is needed for two reasons. The first is to allow for a digital signature, or, an encrypted hash that can be used by someone to authenticate the identity of the sender of a message or of the signer of a document. A digital signature combines attributes of the authentication, integrity and confidentiality building blocks to provide strong authentication and a higher level of integrity.

The second, more familiar function of confidentiality is to keep the contents of a transaction private.

Another security building block is needed to track the day-to-day operations of the corporate network, auditing.
5.5.5 Auditing

Given that there is no such thing as 100 percent security, the next best option is to provide a high level of assurance that an unauthorized event will not occur. In the event that an unauthorized event does happen, an auditing system needs to be in place to provide digital forensics. This is where the fifth security building block, auditing, comes into play. A properly configured audit trail system will, at a minimum, answer the questions, “Who did what, when, how and where?”

Intrusion detection systems (network/host based) and firewalls can tell you that a problem has occurred, but determining the full extent of that intrusion requires in-depth analysis of detailed records. While a properly configured audit trail system will record transaction data, it cannot by itself prove the parties involved, authorization level of participants, improper disclosure of information, and integrity of the transaction. The audit trail is a record of the events that occurred and relies on the correct implementation of the previously mentioned security building blocks and the next security building block, non-repudiation, to provide secure transactions.

5.5.6 Non-Repudiation

The sixth and final security building block is the ability to provide non-repudiation, which cannot occur until the previous five security building blocks are in place. Non-repudiation is the ability of the recipient of the transaction to prove to a third party that the sender really did send the message. This is only half of what is needed to claim full non-repudiation on a transaction. To provide full non-repudiation for both parties, the following steps must be taken:

- All parties must be identified and authenticated,
- All parties must be authorized to perform the functions required,
- The integrity of the transaction content must be intact throughout the entire process,
- Certain transaction information needs to be confidential for authorized users only, and
- The transactions must be audited.

Only when all of the items listed above are correctly pieced together can an organization claim non-repudiation.
5.5.7 Recommended Security Architecture

Network, application, and database auditing will all be used for tracking activities throughout CYFD SACWIS

Figure 5.5.7-1: The Security Building Blocks Recommended for FACTS

Successful projects begin with the fundamental building blocks discussed above. These six security building blocks, when combined, represent the most complete way to ensure a secure transaction.

Our security staff will be reviewing WS-Security, WS-SecurityPolicy, WS-Trust, WS-SecureConversation, WS-Federation, XACML, XrML, XKMS, XML-Signature, XML-Encryption, SAML and the WS-I Basic Security Profile as we further our journey into the world of service-oriented computing.

See Appendix 12.15, Confidentiality Building Block and Appendix 12.16, Non-Repudiation Building Block for a discussion of these building blocks. There are also Reference Materials, (see Appendix 12.4) discussing Security Policy as currently promulgated by CYFD and a White Paper discussing the benefits of PKI technology. The Computer Security Policy by CYFD and the State of New Mexico are the baseline security documents that ITS will use to start documenting their security policy. The Information Security Management Handbook by Tipton and Krause is also an excellent source for covering the entire security landscape.
In addition to implementing the six building blocks, ITS must be vigilant in detecting, inoculating, preventing and protecting their software assets from virus attacks. Since no security systems is 100% perfect, intrusion detection software must also be installed with human follow-ups on security breaches.

The following presents requirements for the ADE/Security module:

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Sign-on User ID and Password</td>
<td>Authentication</td>
</tr>
<tr>
<td>Digital Certificates X.509</td>
<td>External user verification and non-</td>
</tr>
<tr>
<td></td>
<td>repudiation</td>
</tr>
<tr>
<td>LDAP</td>
<td>Authorization and organizational</td>
</tr>
<tr>
<td></td>
<td>structure management</td>
</tr>
<tr>
<td>128 Bit Encryption or better</td>
<td>Confidentiality</td>
</tr>
<tr>
<td>Administration Tools</td>
<td>To improve manageability and maintainability</td>
</tr>
<tr>
<td>Intrusion Detection/Malicious Attack</td>
<td>To identify unauthorized access</td>
</tr>
<tr>
<td>Prevention</td>
<td></td>
</tr>
</tbody>
</table>

The security building blocks will have a technical architecture as described in the graphical depiction in Figure 5.5.7-1. The building blocks are spread throughout the system architecture to protect all points of vulnerability.

5.6 Application Architecture Decisions

This section documents the process that will be used to select the application architecture for the FACTS application. The process consisted of four steps:

- Review the technical requirements in the context of the enterprise,
- Select or define the Application Architecture,
- Select the application development tools and environment, and
- Define the remaining technical architecture components.

It is important to emphasize the reliance on defined requirements as the driving factors in this process. The application architecture selection will not be made by evaluating industry trends. Four possible application architectures currently exist:

- Fat Client Server
- Plump Client Server
- Thin Client Server
- Ultra Thin Client Server
These architectures were evaluated against the requirements to arrive at the selection of an ultra-thin client server architecture for the majority of application components but where necessary utilizing thin client server architecture to support application components where the capabilities of that architecture are specifically required (i.e., forms management).

The primary method of evaluating information will be through a series of technical meetings with State stakeholders. The following summarizes the technical meetings that will be conducted and the decisions that need to be made:

- **Validate Technical Requirements** A meeting with ITS technical staff will be conducted to verify/validate the requirements.

**EXPECTED RESULT:** Technical requirements will be extracted and validated. See Appendix 12.20 for an example of what the requirements may look like.

- **Examine Existing Infrastructure of the State Network**

We will meet with the DoIT Network Infrastructure Team to get details on their current network infrastructure.

**EXPECTED RESULT:** Descriptions of the DoIT Network Backbone, ITS State Office Topology, ITS hardware configuration for a typical remote office, ITS typical data flow and the ITS network hubs in Albuquerque, Santa Fe, Las Cruces, Ruidoso and Clayton were presented.

See Appendices 12.10 through 12.11 for graphical representations of the current status of the CYFD Network.

- **Application Architecture** Workgroup meetings were conducted to produce an Evaluation Criteria matrix. Once this form was completed, another meeting was conducted to evaluate the criteria. This is the result of an exercise by another State which we will use.

**DECISION:** A matrix comparing Fat/Plump and Thin/Ultra-Thin architectures against twenty-one evaluation metrics were created. The Thin/Ultra Thin Architecture was selected. See Appendix 12.7.

- **Weighting the Architecture Evaluation Criteria** In considering the State of CYFD’s FACTS System Application Architecture, specific evaluation criteria will be developed to be applied to the various architecture alternatives. These criteria will be weighted among the alternatives to achieve an overall measurement of the user, business and data services as a basis for architecture selection.

**DECISION:** It was determined that weighting was not required. See Appendix 12.5.
• Single vs. Multi Vendor ADE
Specific evaluation criteria will be developed to be applied to determine whether an Open Source, Single Vendor or Best of Breed approach would better serve the system development effort. These criteria will be weighted to achieve an overall measurement of user, business and data services as a basis for selection.

DECISION: See Appendix 12.6. A Single Vendor Application Development Environment for a sample of the evaluation criteria.

• Thin vs. Ultra-Thin Specific evaluation criteria were developed to be applied to determine whether a Thin or Ultra-Thin Architecture would better serve the development of the system.

DECISION: See Appendix 12.7. The Ultra-Thin Architecture was selected to serve the majority of application functionality. The only reason to implement with Thin Architecture would be due to the inability for the Ultra-Thin to provide the desired functionality.

• Centralized vs. Distributed Application Architecture

We will be evaluating whether a distributed (as in IONA Technologies, Inc) approach makes more sense in our environment versus a centralized architecture as represented by IBM, Oracle and HP.

5.7 Application Architecture Assumptions
Analysis of the Business functions and requirements has yielded the following consolidated, technically interpreted list of these functions and requirements that affect the overall architecture:

- Utilize the existing WAN/Intranet/Internet Infrastructure,
- Easily scalable, portable and reliable,
- Availability (24x7),
- Web-based application delivery,
- Real Time access to existing systems - synchronous, asynchronous and batch,
- Robust graphical user interface (GUI),
- Interface with office automation tools,
- Interface with Microsoft Outlook e-mail application,
- Batch Processing,
- Timely Reporting,
- Ease of deployment internally and externally,
- Ease of development,
- Configuration Management,
- Security,
- Maintainability,
- Remote manageability, and
- Response time.
5.8 Total Cost of Ownership

The most compelling argument for selecting one particular system architecture over another is achieving the lowest total cost of ownership (TOC). In considering the application architecture, the total cost of ownership was both a specific evaluation criterion individually scored and implied in evaluation criteria such as maintainability, manageability and reliability as well as the ease, speed and cost of development. Costs were not specifically developed for each application architecture because there are significant variances between them and well understood by the technical organizations involved in the assessment process. This base of knowledge was relied upon as the basis for the scores received in each category.

Once specific products have been identified a complete analysis will be made with input from this document.

Application Architecture Constraints

No constraints have been identified.
6 FACTS Hardware and Network Architecture

6.1 Hardware and Network Architecture Description
The hardware and network architecture describes the computers and networks on which the system operates and their configurations. Appendices 12.9 through 12.12 provide graphics of the current topology and data flow.

6.2 Hardware and Network Architecture Approach
The primary inputs to the Hardware and Network Architecture are the Requirements Documents and meetings with the technical team from CYFD. The team expressed their wishes to leverage as much of the current infrastructure as possible. The following table provides a list of criteria that Hardware and Network Architecture need to meet:

<table>
<thead>
<tr>
<th>#</th>
<th>Evaluation Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Support Required Functionality</td>
</tr>
<tr>
<td>2</td>
<td>Online Performance and Response Time</td>
</tr>
<tr>
<td>3</td>
<td>Compatibility with Existing IT Infrastructure</td>
</tr>
<tr>
<td>4</td>
<td>Scalability as Workload Increases</td>
</tr>
<tr>
<td>5</td>
<td>Reliability</td>
</tr>
<tr>
<td>6</td>
<td>Maintainability</td>
</tr>
<tr>
<td>7</td>
<td>Manageability</td>
</tr>
<tr>
<td>8</td>
<td>Required Hardware and Software</td>
</tr>
</tbody>
</table>

6.3 Hardware and Network Architecture Selections
This section describes the hardware and network architecture used to deploy the FACTS applications. Furthermore, this section describes the principles to be considered when upgrading workstations and networks over the course of FACTS development and implementation.

Two views of FACTS hardware are provided here. The Hardware View provides a general list and configuration for the computer systems that plays some role in FACTS. It also provides the capabilities of hardware that was purchased. The network view describes and groups the computer systems by location and documents the network configurations used to connect these locations. The network view also provides a general template to be used to collect and document the hardware deployed at a given
6.3.1 The Hardware View
This section documents the hardware for FACTS. Sections 6.3.3-6.3.9 describes the various computer system categories to be used in the Hardware Architecture for FACTS. Appendix 12.18, Server Hardware Evaluation Matrix provides a sample template that will be used to evaluate the hardware alternatives for FACTS.

6.3.2 The Network View
The FACTS project leverages the current network infrastructure of CYFD. The following sections will discuss the topologies of the Children Youth and Families Department’s (CYFD) State Office, and CYFD Remote Offices, all of which will support the FACTS application. At this point, this network has enough capacity to support the requirements of FACTS. See Appendices 12.9 -12.12.

6.3.2.1 ITS Topology
See Appendix 12.9.

6.3.2.2 CYFD Remote Office Topology
A typical ITS Remote Office is depicted in Appendix 12.10, Children Youth and Families Department Typical Remote Office.

6.3.3 CYFD Enterprise Server Overview
Information Technology Services (ITS) is a 7 day per week, 24 hour a day operation providing services to CYFD. On weekends, before 7:30AM and after 5:30PM on weekends and holidays, staff are ‘on call’.

6.3.4 Application Servers
All server equipment should be rack mounted for ease of maintenance.
<table>
<thead>
<tr>
<th>Purpose</th>
<th>Make/Model</th>
<th>Configuration (minimum)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web and Application Server(s)</td>
<td></td>
<td>Currently None</td>
</tr>
<tr>
<td>Database Server(s)</td>
<td></td>
<td>HP 9000</td>
</tr>
<tr>
<td>Web and Reporting Server</td>
<td></td>
<td>Currently None</td>
</tr>
<tr>
<td>Other Servers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WebSense</td>
<td></td>
<td>HP Proliant ML370 Linux server, Dell PowerEdge 2650 Microsoft server</td>
</tr>
<tr>
<td>Intranet</td>
<td></td>
<td>Dell PowerEdge 2500 SUSE Linux version 9 server, 2 Premio servers with SUSE Linux version 9 server.</td>
</tr>
<tr>
<td>Actuate</td>
<td></td>
<td>P Series 660 IBM AIX server, P Series 610 IBM AIX server</td>
</tr>
<tr>
<td>DNS</td>
<td></td>
<td>HP Net Server LC2000 Linux Server</td>
</tr>
</tbody>
</table>

For a list of all of CYFD's Novell servers, see

G:\TSS\Novell\Documentation\Netware 6 Information\NW 6 Planning XLS\Novell Environment.xls

**6.3.5 CYFD Desktop Hardware**

The following summarizes the current desktop hardware throughout the State.
### Purpose

#### Make/Model

#### Configuration (minimum)

**Desktop PC**

Gateway Professional E2300

Pentium 4 Processor 3.0 Ghz, 40GB Ultra ATA100 7200rpm hard drive VX755 17” Color Monitor

### 6.3.6 CYFD File Servers

The following summarizes the current file server configurations throughout CYFD.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Make/Model</th>
<th>Configuration (minimum)</th>
</tr>
</thead>
<tbody>
<tr>
<td>County: File/Print Services</td>
<td>HP and Dell</td>
<td>HP Proliant ML370 Linux server, Dell PowerEdge 2650 Microsoft server</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dell PowerEdge 2500 SUSE Linux version 9 server, 2 Premio servers with SUSE Linux version 9 server.</td>
</tr>
</tbody>
</table>

### 6.3.7 Network Hardware

Details regarding specific equipment and connectivity to CYFD’s infrastructure is included below. Firewall standards can be referenced in Appendix 12.21: ITS Firewall Standards.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Make/Model</th>
<th>Configuration (minimum)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firewall</td>
<td>Cisco PIX 515E-46-BUN (two- one as a fail-over)</td>
<td></td>
</tr>
<tr>
<td>Switches</td>
<td>Cisco 1912EN,1924EN, 2940-8TF-5,2948,6506-S1, 2950E-24-E1,3550-12G,3550-24FX-SMI,3550-48EMI,3508,3548,3524-XL-EN.</td>
<td></td>
</tr>
</tbody>
</table>
See the Access Database below for all of CYFD’s network equipment:
G:\Share\TSS\TeleCommDB\TeleComTracking2.mdb

6.3.8 State Office/County Network Printers
The following summarizes the types of printers currently used throughout CYFD.

6.3.9 Miscellaneous Hardware

Cabling Type
Children Youth and Families Department currently utilizes DoIT Standard wiring where applicable.

Network Protocol
Wide Area Network - TCP/IP. Local Area Network - TCP/IP

6.4 Hardware and Network Assumptions
FACTS employed two strategies to avoid performance degradation due to insufficient hardware and network capacity. The first strategy utilized the application and network model to predict hardware and network requirements early in the development cycle. This allowed potential performance problems to be addressed early in development, and avoided performance problems on the part of the application. This strategy also required monitoring of performance through development.
However, FACTS will be one application of many using the network and the WAN network connections. The second strategy was to monitor network usage closely and address increases in bandwidth requirements quickly. As FACTS is deployed, the network and application bandwidth and computational load will be closely watched. Any performance issues with FACTS itself (beyond the resource requirements expected) can be recognized and addressed quickly. If the bandwidth usage increases past 50 percent peak, ITS will either address network usage problems through policy (such as controlling what applications use the network, and at what times) or ITS will work to increase the network capacity.

Another key assumption is FACTS use of the CYFD Wide Area Backbone to connect the various locations involved in building and deploying FACTS.

The current system management tools are used by FACTS. Most of these tools are provided for remote access to manage:

- end-user support functions,
- servers,
- discovery of hardware and software, and
- electronic software delivery.

ITS also utilizes tools for asset management and software distribution.

Current performance management tools are used. Tools are used to analyze performance of routers/switches and circuits. These tools help determine impact analysis of traffic on servers, partitions, replicas, and containers. Data backups are performed using the Omniback software. These back ups are performed nightly, monthly, and yearly for the FACTS Database. Tape backups are currently used for the 33 county offices, 5 regional offices.

6.5 Hardware and Network Constraints

All locations involved in FACTS have high bandwidth connections. There are currently no other identified constraints for the existing platform.
7 FACTS Software Architecture

The software architecture describes the software layers and components and the interaction to implement the business architecture. This includes operating systems, third-party software, system management services, database management systems, development tools, testing software, and standard interfaces such as WS-*; CORBA and OLE. Also, included are the application, interfaces, user interface, and common services software.

In FACTS there will be several software architectures one for each unique level of the system. All are interrelated and often hard to determine just what part is the driving force. In the design of the FACTS’ system, the Application Development Environment was the first technical architecture to be investigated. See Appendix 12.14, ADE Criteria Issues/Questions, for typical questions that should be asked when considering an ADE.

Figure 7-1: Traditional Software Layer Diagram

Figure 7-1 shows an example of a traditional software layer diagram. Software layers only communicate directly with other components within the same layer or in layers that share a boundary. This layered approach serves to insulate components in one layer from changes in another. As the design progresses, this high-level view will be decomposed into specific software components (such as objects, structured modules, and business services) and the interfaces between them. The stair-stepped layers...
indicate that components in one layer can access several lower layers. In Figure 7-1, the Application Business Functions make direct calls to several lower levels.

This diagram illustrates generic software layers, but the chart can be expanded to illustrate chosen programming languages and other enabling software.

In the diagram, the bottom two layers (Operating System, DBMS, Middleware and Network Protocol) would often be referred to as the Run Time or Execution Environment. The remaining layers would be referred to as the Application Development Environment (ADE). The two environments are dependent on each other and in fact often the specification of one drastically impacts or even forces a decision for the other. When one looks at a n-tier environment, it is possible there might be multiples of a number of the environments.

In the FACTS environments it was decided that a web enabled, SOA compliant architecture be implemented for the delivery of functionality to the end users.

### 7.1 Software Architecture Description

The components are visually represented in Figure 7-1. The following defines these components:

**User Interface Services**
These consist of the parts of the application that the end user sees as the ‘application’ or business process. One often views these as the visual tools and standards that the developers use to produce the application. Included in this area are things like drag and drop, folders, tabs, drop down boxes, views and colors. When developing software these are some of the functions of the GUI (graphical user interface) that the development environment provides along with the local enhancements and standards that are necessary to provide the ‘look and feel’ of the application.

**Application Business Services**
These are the start of the components that are unseen by the end user but are specifically tailored to the specific business functions required by ITS.

**Common Business Services**
These continue the unseen parts. These refer to common business service that are independent of ITS.

**Database Access**
These components allow for consistent access to the databases and other data interfaces that the application requires. Often the end users believe that the database is seen but, what is really seen is a view of the database displayed via the Graphical User Functions.

**DBMS (Database Management System)**
The database management system is the storage vehicle for the actual data that is used by the application.

**Middleware**
This includes the part of the application that communicates with other systems in a standard way. The Registry and Repository, Enterprise Service Bus (ESB), the SOA Management Tools would be found in this layer.

**Network Protocol**
This is the physical signaling among the parts of the application including the end user.

**Operating System**
This is the fundamental basis for the layers built upon it. Thus, software/code that runs the hardware is chosen for its stability, consistency and adherence to standards.

These components comprise the base system. In addition, there are a number of other parts that are necessary in a complete system. Figure 7-1 describes the end user machine.

The software architecture describes the software layers and components and how they interact to implement the business architecture.

### 7.1.1 Software Architecture Detailed Design

The FACTS specific design for the integration of software architecture components will be detailed in the FACTS Entity Relationship Diagrams (ERD).

### 7.2 Software Architecture Approach

The primary inputs to the Software Architecture were the CYFD Requirements Definition and the discussions and meetings with the program and technical teams from CYFD.

The decision to purchase and install software should be driven by an immediate need. As FACTS development continued, all decisions for the purchase and deployment of software in support of FACTS should be documented in this section.

Software must be installed on a suitable computer system, and fit within the network architecture for CYFD. The Software Architecture is described from the viewpoint of software on hardware to better represent this view.

The Software Architecture requirements list is a subset of the Application Architecture Evaluation Criteria. The table below identifies the Software Architecture Evaluation Criteria.

<table>
<thead>
<tr>
<th>#</th>
<th>Evaluation Criteria</th>
</tr>
</thead>
</table>
1. Support Required Functionality
2. Online Performance and Response Time
3. Batch Processing
4. Timely Reporting
5. Ease of Use and Robustness of the GUI
6. Ease of Interfacing to Other Systems
7. Ease of Deployment to private agencies
8. Ease of Deployment to CYFD Sites
9. Integration with Desktop Applications
10. Ease, Speed, and Cost of Development
11. Security
12. Compatibility with Existing IT Infrastructure
13. Reliability
14. Maintainability
15. Manageability
16. Cross-Platform Support
17. Ease of User Support
18. Required Hardware and Software

The Software Architecture discussions, involving the CYFD technical team and the Quality Assurance technical team will occur through meetings and discussions. During these meetings, a decision affecting the final work product will be made. It will be determined for the purpose of this document that only the capabilities of Application Development Environments necessary for SACWIS implementation would be investigated as opposed to selecting a specific ADE from a specific vendor. A related decision is that for an ADE environment, a single vendor/partnership would be preferable to a collection of disjointed products that the developer would need to weave together.

7.3 Software Architecture Selections
To determine the appropriate Application Development Environment (ADE), to develop the FACTS application, possible ADE approaches were considered:
Best of breed (Multi-vendor)
Single large company or partnered alliances (Single-vendor)
Open Source

When the decision is made to purchase the ADE, it will be Commercial Off-The-Shelf (COTS) software. The COTS Development Software must:

- Support software specifications in a visual manner rather than textual, a descriptive manner rather than procedural, and a natural manner rather than cryptic,
- Provide a full range of functionality within an ultra-thin web-enabled application server paradigm for applications with complex business rules operating on complex data structures,
- Support the creation of all Windows’ presentation idioms,
- Support static and dynamic software analysis,
- Support synchronization of development activities across multiple environments,
- Comply with industry standards for affecting interaction among software modules (such as DLL, DDE, and OLE),
- Interact with repository managers, data dictionaries, and configuration management utilities,
- Not maintain its own proprietary database,
- Result in well-partitioned software components of manageable size which efficiently use memory, mass storage and CPU,
- Accomplish server communications through SQL, RPC, Web Services or transaction processing monitors, and
- Have a significant market share and be provided by an organization in good fiscal health.

The above is a generalized list of capabilities required for the software. For detailed requirements, see Appendix 12.20, Technical Requirements for FACTS.

7.4 Software Architecture Decisions
When one looks at the architectural decisions that are made, the most important one is the decision to use a web enabled, SOA compliant architecture. Some of the critical features needed for the ADE are:

- User interface controls,
- Wizards to assist in development and configuration,
- Explorer function to locate and identify the components,
- Scripting controls,
- Business functions availability, and
- Property manager for objects.
7.5 Software Architecture Assumptions
The assumptions made about the software architecture are few. The assumption has been made that all required functionality can be developed with the ultra thin client model.

7.6 Software Architecture Constraints
At this time there are no constraints on the software architecture.

7.7 Configuration Management
Questions concerning the features necessary for configuration management will be included in the ADE questions sent to the potential vendors. The critical ones deemed necessary are:

- The ability to control the configuration at the version, project, object and enhancement level, and
- The ability to deal with the promotion of code/components from the unit, to system, to system test, to pre-production, to production, and finally to expired.

The following presents requirements for the ADE/Configuration Management module:

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project and Object Versions Maintained</td>
<td>To support the development process.</td>
</tr>
<tr>
<td>Coordinated Promotion of all Application</td>
<td>To support the testing and implementation</td>
</tr>
<tr>
<td>Components</td>
<td>process.</td>
</tr>
<tr>
<td>Check In/Check Out</td>
<td>To support the development process.</td>
</tr>
</tbody>
</table>

7.8 Software QA Tools
While the addition of effective testing tools to detect and help resolve both testing and production faults (especially the ones that cover multiple systems) are helpful, the most important criterion for selecting a QA Tool is its ability to assist in performing regression testing when a new or changed function is added to the application.

The following presents requirements for the ADE/QA Tools module:

<table>
<thead>
<tr>
<th>Capabilities Required</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression Testing of a Browser-based Application</td>
<td>To support iterative development and testing.</td>
</tr>
<tr>
<td>Central Repository</td>
<td>To improve maintainability</td>
</tr>
</tbody>
</table>
8 FACTS Data Architecture

The Data Architecture has two sections: the software components that define how the data required to support the system will be implemented and how data are accessed, partitioned and distributed across the network; and the Logical Data Model that consists of a high-level Entity/Relationship Diagram (ERD) for all of FACTS, and a detailed data design.

8.1 Data Architecture Description

The data architecture addresses the physical design, structure, and implementation of the persistent (not transient) data needed for the system's functionality and operations. The business architecture provides a conceptual view of needed data via object models or logical data models. The data architecture addresses the physical structure, location and distribution of the data. It addresses supporting different usage such as transaction versus ad-hoc queries and analysis. Once the Requirements Definition Sessions are completed, a logical model of the system will be constructed. As the Detailed System Design approaches completion, the logical model will migrate to a physical data model. See G:\ADS\ERDs. Figures 8.3.3-1 and 8.3.3-2 depict, at a very high level, data to be accessed from FACTS.

8.2 Data Architecture Approach

The approach for the first section of the Data Architecture is similar to that for the Hardware and Network Architecture. The decisions are driven by meetings with ITS/CYFD Domain Experts. There were no specific meetings conducted to address the Data Architecture. The logical data model will be delivered with the Conceptual System Design and the physical data model with Detailed System Design.

The Data Architecture requirements, however, can be gleaned from the traceability matrix below. Note that the following is a subset of the overall requirements evaluation criteria that were measured in determining the Application Architecture. These requirements were the yardstick that were used in selecting the RDBMS, Sybase Adaptive Server for FACTS. We are currently evaluating PostgreSQL. We have also broached IBM to provide us with some preliminary information on DB2.
<table>
<thead>
<tr>
<th>#</th>
<th>Evaluation Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Support Required Functionality</td>
</tr>
<tr>
<td>2</td>
<td>Online Performance and Response Time</td>
</tr>
<tr>
<td>3</td>
<td>Timely Reporting</td>
</tr>
<tr>
<td>4</td>
<td>Ease of Deployment to private agencies</td>
</tr>
<tr>
<td>5</td>
<td>Ease of Deployment to CYFD Sites</td>
</tr>
<tr>
<td>6</td>
<td>Security</td>
</tr>
<tr>
<td>7</td>
<td>Scalability as Workload Increases</td>
</tr>
<tr>
<td>8</td>
<td>Reliability</td>
</tr>
<tr>
<td>9</td>
<td>Maintainability</td>
</tr>
<tr>
<td>10</td>
<td>Manageability</td>
</tr>
</tbody>
</table>

**8.3 Database Management Systems**

This section specifies the capabilities a RDBMS should implement. The data modeling tool that will be used is also specified here.

**8.3.1 Server Hardware**

See Section 6.3.4 Application Servers.

**8.3.2 Software**

For Data Analysis, the FACTS project will use PowerDesignor or Rational Rose to contain information about entities, and the linkages between them. It also used the data modeler for building and maintaining the ERDs. There are, however, several database products that have the ‘industrial strength’ and marketplace acceptance to accommodate the application needs:

- DB2
- Oracle
- Microsoft SQL Server
- Sybase Adaptive Server
- Possibly mySQL, PostgreSQL Open Source RDBMSes:

DB2, Oracle and Sybase are currently the database engines used/planned for several SACWIS implementations. The above products are also sold by companies that currently have the financial health needed for long term viability. Sybase's Adaptive Server Enterprise was the selected RDBMS for FACTS.
8.3.3 Data Access

Generally, data should be accessed across platforms ONLY through the mechanisms normally provided on that platform, and not through direct input/output. Use of standard mechanisms simplifies data management, system performance management, capacity management and problem determination because all data access is clearly visible to Operations and Support staff. In addition, new transactions may be created as needed on a familiar platform, and the data access can be managed and tuned for that environment. This approach allows both the legacy and new functions to use the identical rules and code for reading and updating data, ensuring that the same rules are applied to the data in all cases. See Figure 8.3.3-1, Data Access for a breakout depiction of the data access component of the application architecture.

There will be no direct access by the individual users to the FACTS Database. User access will be accomplished via middleware or a customized I/O layer that will facilitate reading, updating, deleting and writing data to/from the database. See Figure 8.3.3-2, Uniform Data Access Method for an overview graphic which depicts how data will be accessed. See Section 5.3.4, Data Access for the Capabilities Required.

**Figure 8.3.3-1 Data Access**
There will be no direct access by the individual users to the FACTS Database. User access will be accomplished via middleware or a customized I/O layer that will facilitate reading, updating, deleting and writing data to/from the database. See Figure 8.3.3-2, Uniform Data Access Method for an overview graphic which depicts how data will be accessed. See Section 5.3.4, Data Access for the Capabilities Required.

Figure 8.3.3-2. Uniform Data Access Method

Features required by the RDBMS needed to support the security aspects of the system are:

- Triggers for auditing
- Views to implement business logic
- Grants for tables and views
FACTS users access the operational data through 'ultra thin client' interface which provides authorization mechanisms implemented through the security schema and assignment and approval functions to protect the data from unauthorized access. To support report generator/writer tools and power users who require direct access, the FACTS database administrators will implement a “Query Only” database protected only by the RDBMS features. This enabled the Quality Assurance process and other qualified users to obtain needed statistical information from FACTS.

Similarly, the FACTS data warehouse (when implemented) will be available in a read only form to protect the information contained in it. However, Quality Assurance users and other qualified users will have the capability to define additional transformation rules to the extract process to generate new information into the OLAP database. See Figure 8.3.3-2, Uniform Data Access Method for an overview graphic which depicts how data will be accessed. See Section 5.3.4, Data Access for the Capabilities Required.

8.4 Data Warehouse
This section has been moved to the Data Warehouse Design document (which has not been started as of the date of this document). Please see this document for all information regarding the FACTS Data Warehouse.

8.5 Entity-Relationship Diagram
An Entity Relationship Diagram (ERD) is a tool for designing databases that shows the kind and organization of the data to be stored in the database. It depicts the things of importance in an organization (entities), the properties of those things (attributes) and how they are related to one another (relationships). The ERD is independent of any data storage or access method.

8.6 Data Retention
Archiving rules will have to be defined as part of the Archiving Detailed Design Topic Paper. Data impingement rules will also have to be defined in the Expunge and Purge Detailed Design Paper.

8.7 Backup and Recovery
Two areas of the FACTS system architecture require backups be made to ensure the software and data can be recovered. The servers need complete disk backups made by the utility designed to support the operating system environment and the operational database must be imaged by the RDBMS provided utility before and after the batch update cycle (see Section 8.8). The backup processing is completed twice each operational day with transaction dumps each hour.

The server backups and the second database image (made after the batch update cycle) is stored offsite to support disaster recovery procedures. Daily backups can be
kept for one prior week and weekly backups for three months and monthly backups for six months. A copy of the prior week’s backups is kept onsite to support non-disaster recovery scenarios.

Recovery processing for the operational database uses the RDBMS provided utilities to utilize the database log and most current backup reducing any lost data to a minimum. Recovery on a server for a disk crash or other malfunction is performed by the utility designed to support the operating system environment and needs to be accomplished for all disks in the file system.

### 8.8 Batch Processing Schedule

The FACTS Operational Procedures document serves as the blueprint for planning scheduled activities within FACTS. There are three types of activities that will be scheduled for FACTS:

- Batch Programs,
- Batch Interfaces, and
- Scheduled Reports.

The purpose of the FACTS Operational Procedures document is to assist in sequencing these scheduled activities and to ensure that dependencies between these activities are accounted for. The Operational Procedures document includes the batch programs, interfaces and schedules reports that have been documented as part of the Detailed System Design (DSD) deliverable. Please refer to the DSD for more detail on the individual activity. See:

- G:\OPS\Operations_Guide\Operations_Guide.doc
- G:\OPS\Operations_Manual\opsmanual.doc
- G:\OPS\Policy_Manual\BITPolicies.doc


The following legend is intended to assist in reviewing of the Operational Procedures document:

<table>
<thead>
<tr>
<th>Column Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script #</td>
<td>The FACTS Operational Procedures document groups the scheduled activities in functionally meaningful groups called “scripts.” For example all batch programs that are needed to generate the AFCARS transmission file can be grouped in one script and run all at once. Each script in the Operational Procedures document is identified with a unique script number. This script number is documented in the “Script #” column.</td>
</tr>
<tr>
<td><strong>Column Label</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Script Run Sequence</td>
<td>The Script Run Sequence indicates the order in which the scripts will be executed. The sequence of a script execution is important since it allows us to enforce dependencies between scripts. For example, Script 4N cannot be executed before Script 3N is successfully completed.</td>
</tr>
<tr>
<td>Script Name</td>
<td>Script Name indicates the name of the script that groups the related programs together.</td>
</tr>
<tr>
<td>Program #</td>
<td>In each script, there are one or many programs. A program can be a batch program, a batch interface or a scheduled report. Each of the programs referenced in the Operational Procedures document are defined further in their respective sections as part of the DSD deliverable.</td>
</tr>
<tr>
<td>Program Name</td>
<td>Program Name indicates the name of the program (batch program, batch interface or scheduled report) as described in the DSD deliverable.</td>
</tr>
<tr>
<td>Program Run Sequence</td>
<td>The Program Run Sequence indicates the order in which the individual program within a script will be executed. The sequence of a program execution is important since it allows us to enforce dependencies between programs. For example, FACTS Bank Recon interface cannot be executed before Disbursements Run batch is successfully completed.</td>
</tr>
<tr>
<td>Batch/Interface/Report</td>
<td>Indicator to show the type of program: • B – Batch • I – Interface • R – Report</td>
</tr>
<tr>
<td>Topic/Report Name</td>
<td>For Batch Programs and Interfaces, this column depicts the name of the DSD design paper that includes the referenced program. For Reports, this column depicts the name of the report as included in the Reports section of the DSD deliverable.</td>
</tr>
<tr>
<td>Special Day</td>
<td>Indicates whether the program is required to run on a special of the week, month, etc. For example, “last Thursday of every month”</td>
</tr>
</tbody>
</table>
9 Governance

Governance describes those processes established by an organization to determine who is empowered to make certain decisions. It is a framework which provides an infrastructure for creating, communicating, and enforcing corporate policies across the organization. Governance is usually viewed as a function of senior management while the execution of the governance framework is the responsibility of managers. For example, governance determines the decision rights for how much the enterprise invests in technology. IT management is responsible for deciding the actual amount invested in a given year and the areas in which the money is spent. In this chapter, we will discuss overall IT Governance and then address a framework for Service Oriented Architecture (SOA) Governance.

9.1 IT Governance

Governance, from an information technology (IT) perspective, represents the processes established by an organization to determine:

- What decisions must be made to effectively manage the assets of IT?
- Who is empowered to make these decisions?
- How will these decisions be made, measured and monitored?

It is the policies, principles, standards, procedures and processes that are used by the organization to measure and control the way these decisions are implemented. It deals with the management and control of IT assets, people, processes and infrastructures, as well as the manner in which the assets are managed and procured. It is through governance that IT activities are aligned with the goals of the Agency and the goals of the State. Through the funding process, IT decisions are prioritized, measured, controlled and executed. The IT investment decision is the most significant of the decisions about the procurement of an IT asset. It includes:

- How much to spend?
- What IT asset should be purchased?
- How to create a balance between competing Agencies with similar priorities especially when there are limited funds to expend?

A proper governance strategy will manage an IT asset through its entire lifecycle. For hardware, this would include the acquisition phase, the maintenance process through the retirement and subsequent replacement of the hardware. For software, this would include the requirements specification, analysis and detailed design phases through the development, deployment and subsequent retirement and replacement of the software.

In building a governance process, it is important to also consider the ‘soft’ skills required. Communication and collaboration are essential ingredients to facilitate a functioning process.

The following graphic, Figure 9.1.1, is the proposed governance structure for New Mexico’s IT Enterprise Architecture (ITEA). Please see “State of New Mexico Framework for Enterprise Architecture” pages ITEA-11 through ITEA-19 for a discussion on the proposed governance process.
Note: With the changes brought on by the new Department of Information Technology (DoIT), Figures 9.1.1 through 9.1.3 will be updated when DoIT documents its processes.

Enterprise Architecture Program Governance
Figure 9.1.1 Enterprise Architecture Program Governance

The following two graphics present overviews of the Project Certification Committee (PCC) and IT Commission (ITC) Certification Governance Processes promulgated by the State CIO. The second graphic provides more process flow information for the ITC, PCC and Architectural Committee (AC).

Figure 9.1.2 presents a graphic of the IT Project Certification Process which flows from an Agency request to the Office of the Chief Information Officer (OCIO) review.

Figure 9.1.2 IT Project Certification Process

![IT Project Certification Timeline and Gates](image)

**KEY**

- Input Documentation
- Project Phase
- PC Approval Gate
- PC Approval Gate & Agency Presentation

*Note: Upon PCC Approval and Certification, DoIT will issue a letter to DFA authorizing the release of funds either:

1. Release of Funds (No requirements or contingencies)
2. Release of Funds with Requirements- The Agency will receive their released funds, however, the Agency must complete the requirements otherwise the project will be halted.
3. Release of Funds Contingent upon completed tasks- The Agency must first complete the required contingencies prior to the release of funds.*
In Figure 9.1.3, we are exposed to the Information Technology Committee Review, the Project Certification Committee (PCC) Review and the Architectural Committee Review.

Figure 9.1.3 IT Certification Process
See Figure 1.2-3 “State of New Mexico Enterprise Architecture Components. Governance is the architectural base that the other components are built on. It is the processes and oversight geared at ensuring that the architecture actually delivers the value promised. The primary emphasis from the DoIT in the governance arena is project planning and funding.

Within the Children Youth and Families Department (CYFD), the high level governance body is the Senior Staff - directors of the Program Areas and Program Support areas, deputy secretaries and the Cabinet Secretary. This body approves the overall direction and mission of the Agency. The Information Technology Change Control Committee (ITCCC) implements the specified policies.

![Diagram of ITCCC structure]

Figure 9.1.4 – Information Technology Change Control Committee

ADS– Application Development Section
The Information Technology Change Control Committee (ITCCC) meets on a monthly basis to discuss the various on-going IT projects and their statuses. All Change Requests and Incident Reports involving FACTS are evaluated, prioritized, and coordinated by the ITCCC.

The Committee is made up of representatives from each division of CYFD, and is chaired by ITS ADS Management. However special sessions can be called to address critical issues. It also discusses future IT initiatives attempting to balance the IT resources with the work required. When there are unresolvable resource constraints, the issue(s) will be elevated to the CIO who brings it before the Senior Staff (IT Steering Committee) for resolution. See Appendix 12.23, “State of New Mexico Children, Youth and Families Information Technology Change Control Committee Charter” for specifics on why the committee was created and what it does.

**Information Technology Change Control Committee Checklist**

Committee Chair (ITS):
- Appoint a recording secretary from ITS to record official meeting minutes.
- Update the FACTS Change Control list with new Change Requests and Incident Reports that affect FACTS.
- Schedule a meeting of the ITCCC.
  - Members include one representative from each CYFD Service Area.
  - If a representative cannot attend, an alternate may be designated.
  - Meetings normally occur once a month.
  - Special sessions may be called to address critical issues.
  - Arrange a time, place, and facility for the meeting.
- Prepare an agenda; deliver the agenda and the updated FACTS Change Control list to members one week before the meeting.

- Preside over the meeting.

- Do not vote on issues except to break tie votes.

- Report the results of the meeting to the IT Steering Committee.

**ITCCC Members:**
- Review the agenda and the FACTS Change Control list before the meeting.

- At the meeting, review each change on the list, evaluating the following:
  - Criticality, priority.
  - Difficulty of the change (based on ITS estimates).
  - Any applicable federal and/or state requirements.

- Determine which of the categories each change falls into:
  - Emergency modifications.
  - Batch modifications to be implemented shortly.
  - Online modifications to be included in the next release.
  - Changes to be deferred to a future date.

- Vote on any issues that are not unanimously agreed to.
  - Each division’s representative casts one vote.
  - If a representative is absent, the designated alternate may cast that division’s vote.
  - The ITCCC chair may only vote in case of a tie vote.

Within the ADS unit, governance is administered throughout the software development lifecycle (SDLC),

The Applications Development Section (ADS) makes changes to the FACTS system. Changes to batch processes are described in the **Batch Methodology** process description. Changes to online processes follow the System Development Life Cycle described here. This methodology consists of ten phases, with each phase consisting of specific activity and producing specific deliverables. The **Release Strategy** process description details the implementation of the changes in the production environment.
Figure 9.1.6 System Development Life Cycle Checklists

Description of Phases

- Phase 1 - Overview
- Phase 2 – System Development Phase Plan
- Phase 3 – Organization Plan
- Phase 4 – Test Plan
- Phase 5 – Change Control Plan
- Phase 6 – Documentation Plan
- Phase 7 – Training Plan
- Phase 8 – Project Review and Reporting Plan
- Phase 9 – Implementation Plan
- Phase 10 – Resources and Deliverables

**Phase 1 – Overview**

The objective of Phase 1 is to summarize the entire project. Please refer to the document *Project Plan – Overview* for a detailed description of this phase of the project.

- Develop a Project Plan.
- Include each of the Phases in the Project Plan.

**Phase 2 – System Development Phase Plan**

This section of the plan defines the System Development Life Cycle for the project. This phase accounts for the majority of the time spent on the project. The System Development Life Cycle is broken down into phases; each phase is outlined in the System Development Phase Plan. Please refer to the document *Project Plan – Phase 2 – System Development Phase Plan* found...
in Appendix 12.24 for a detailed description of this phase of the project, including primary and secondary objectives and deliverables.

- Definition Phase – Together with the users, define the purpose of the project and the business rules required to implement the project.

- Design Phase – At this point of the project, analysis is complete, the Project Plan has been created and the project teams are defined. The objectives of this phase are to translate the user requirements into detailed system requirements, refine the Project Plan, and receive user approval of system requirements before system design begins.

- Software Development Phase – The approved Requirements Matrix is used to create the detailed System Design document. The Online System Design Document Template can be found in Appendix 12.24. Upon user approval of the System Design document, coding begins. Please see Online Software Coding Standards and Naming Conventions in the “Operations Manual”.

- System/Regression Test Phase – ITS Business Analysts rigorously test FACTS to ensure that the software functions correctly. Individual test cases are used to test both new and existing functionality. All components of FACTS are tested.

- Acceptance Testing Phase – This phase provides the user with the opportunity to test the system modifications and verify that the system functions correctly and according to design specifications.

- Implementation Plan Phase – This phase implements the project in the production environment. Refer to the Release Methodology process description for a detailed description of implementation.

### Phase 3 – Organization Plan

This section of the plan defines each team and the assigned roles and responsibilities during each phase of the project.

- Identify the teams for the project.
- Select the appropriate members for each team.

### Phase 4 – Test Plan

This section of the plan describes the procedures and team responsibilities for the various testing phases of the project.

- Develop a skeletal Test Plan as soon as the requirements are approved.
- Develop test cases for each item in the Requirements Definition Matrix.
- Develop regression testing objectives.
Phase 5 – Change Control Plan
This section of the plan defines the types of change that occurs during the project and the mechanism for managing the change.

- Develop a Change Control Plan.
- Set up a change control mechanism.

Phase 6 – Documentation Plan
This section of the plan describes all of the documents that are created during the project. Not all documents are known at the beginning of the project so this section may be amended throughout the project to represent the documentation needs of the project.

- Develop a list of documents to be developed.
- Develop a list of documents to be modified.

Phase 7 – Training Plan
This section of the plan defines the different types of training that are required during the project.

- Develop a Training Plan.
- Arrange for training.

Phase 8 – Project Review and Reporting Plan
This section of the plan addresses the procedures for formally reporting the status of the project.

- Develop a reporting schedule.
- Create a reporting form.

Phase 9 – Implementation Plan
This section of the plan discusses the procedure for implementing the project in the production environment.

- Refer to the Release Methodology process description for details.

Phase 10 – Resources and Deliverables
This section of the plan assembles, in one place, the resources, schedules, and deliverable items outlined in other phases of the plan.

- Refer to the list of documents identified in Phase 6.
- Gather the documents into one folder on the G: drive.
- Assemble hardcopy documents into a release binder.
**Batch Methodology**

The Applications Development Section is responsible for the development and maintenance of all software associated with the FACTS system. Batch maintains the jobs that run independently (usually overnight). Incident Reports received by Help Desk sometimes indicate an error in a batch process that must be corrected. Various users submit requests for modifications or new functionality (e.g., reports). Batch personnel must review any changes to the FACTS database in order to evaluate the impact on existing batch jobs. All situations follow specific procedures described here.

![Diagram](image)

**Figure 9.1.7 Batch Methodology Checklists**

**Incident Service Reports**

- Log the Incident Report received by Help Desk and pass on to Batch.

- Assign the incident to a Batch staff member.
Analyze the incident to determine what software modification is needed.

Add an addendum to the current design specifying the change. (See the “Operations Guide” – Batch System Design Document Example).

Create another version of the design and move the current one to U:\Factsbat\Version Control\<filename.vx>

Move the current version of the software to U:\Factsbat\Version Control\<filename.vx> where vx is the same extension as the design.

Submit the design document to an ADS Business Analyst for review and approval.

Once approval has been received from the Business Analyst, submit the modifications to users for approval. Note: no software will be modified until users sign the Software Modification Authorization.

Modify the software according to the addendum to the design.

Test the software (see the Quality Assurance Testing checklist).

Save the final set of test results on G:\Batchtest.

Submit the test results to users.

If users request additional modifications directly related to the incident, return to the design modification step.

If users sign the Software Certification proceed to implementation.

Implement the change (up to 1 week)

- Develop or modify the script
- Test the script on the test server
- Develop or modify the operating instructions. Note: if any problems are discovered here, return to design modification, software modification, and/or software testing.
- At the time of the software move, if the software updates the database, complete the following process:
  - Go to G:\ADS\Batch\Cost Allocation folder and access the Software Rating.xls document
  - Create a new row using the current date and add on the number of new lines, number of new transactions, and number of new table hits.
  - Note what the Software Rating was before the rows, table hits, and transactions were added. Change the Software Rating, if necessary, using the rating methodology rules displayed at the top of the spreadsheet.

Set up the Production Run of the software.

- Prepare an e-mail message
  - Send it to Operations Manager
  - Subject line is Production Test
- Attach a completed Production Move Request Form, a Job Run Request form, and if appropriate, modified operating instructions
- Follow up to verify correct results after the CRON scheduler runs the job.

**Change Requests**

Some Change Requests require modifications to existing code, others require new code. When developing new code, be sure to follow the standards outlined in the “Operations Guide” – **Batch Software Coding Standards and Naming Conventions**. Occasionally, a Change Request is for modifications to the data; for example, rate codes or amounts. See the checklist for **New Service Rates and Spending Limits** listed below.

- Assign the Change Request to a Batch staff member.
- Meet with users to define requirements for the change.
- Develop or modify the design document according to the requirements. (See the “Operation Guide” – **Batch System Design Document Example**).
- Rename the version of the design document and move it to `U:\Factsbat\Version Control<filename.vx>`
- Rename the current version of the software and move it to `U:\Factsbat\Version Control<filename.vx>` where vx is the same extension as the design.
- Submit the design document to an ADS Business Analyst for review and approval.
- Once approval has been received from the Business Analyst, submit the modifications to users for approval. **Note: no software will be modified or developed until users sign the Design Certification.**
- Develop the software to satisfy the design requirements. If there are several versions of the design, there must be a software version for each design.
- Test the software and fine-tune the code (see **Quality Assurance Testing** checklist).
- Save the final set of test results on `G:\Batchtest`.  
- Submit the test data to users:
  - If users request additional modifications that affect the design, return to the design modification step.
  - If users request minor modifications that do not affect the design, repeat the software modification and testing steps.
  - If users sign the **Software Certification** proceed to implementation.
- Implement the change (2 weeks)
  - Develop or modify the script
  - Test the script on the test server
- Develop or modify the operating instructions. **Note: if any problems are discovered here, return to design modification, software modification, and/or software testing.**

- At the time of the software move, if the software updates the database, complete the following process:
  - Go to G:\ADS\Batch\Cost Allocation folder and access the Software Rating.xls document
  - Create a new row using the current date and add on the number of new lines, number of new transactions, and number of new table hits. For new software, create a row with the required information.
  - Note what the Software Rating was before the rows, table hits, and transactions were added. Change the Software Rating, if necessary, using the rating methodology rules displayed at the top of the spreadsheet.

- Set up the Production Run of the software.
- Prepare an e-mail message
  - Send it to Operations Manager
  - Subject line is **Production Test**
  - Attach a completed Production Move Request Form, a Job Run Request form, and if appropriate, modified operating instructions
  - Follow up to verify correct results after the CRON scheduler runs the job.

**Quality Assurance Testing**
Several methods may be employed to verify that the code is performing correctly.

- Modify the source code to display key information during execution.

- Compile the source code and run the executable noting the displays.

- Construct database queries to verify the displayed results.

- Remove displays and any other testing aids from the source code and recompile.

- Develop a scenario with a small test case where each result can be calculated by hand; for example, a small county with few members to be counted.

- Run queries directly against the database to verify the results of the test run.

- Save the final set of test results on G:\Batchtest.

- Contact the requestor and/or other key users to do **User Acceptance** testing.

**User Acceptance Testing**

- Once Batch staff has determined that the change has been correctly implemented, notify the Requestor and/or other key users that the code is ready to be tested.

- Give a sample report or data extract to the requestor.
- The requestor compares data to scenarios in the database to determine validity.

**Database Modification Implementation Plan**
- Examine the database changes to identify:
  - New fields added to existing tables
  - Referential integrity changes
- Identify batch programs that access any of the affected tables.
- Examine each identified source code to determine whether the code will be affected.
- Follow the same steps as in **Change Requests** to modify affected programs.
- Coordinate with Online to test the modified program with the modified database.
- Once **User Acceptance** testing has been completed, follow the **Software Migration** checklist.

**Software Migration to Production**
- Prepare an e-mail as follows:
  - Send the e-mail to the Operations Manager
  - E-mail subject line should be the Change Request number followed by a short description (e.g., CR#194 – CPS Reports by Zip Code)
  - Attach the appropriate documents as described here.
- A **Production Move Request** form is required; see the Forms Reference at the top of this process description.
- A **Job Run Request** form is needed to set up the first production run. See the Forms Reference at the top of this process description.
- **Operations Manual Documentation** must be attached if the code is new.
- **Operations Manual Documentation** should be attached if run instructions and/or parameter references must be changed.
- See G:\Ops_Manual\opinst.doc for detailed instructions.

If the job is an “On Request” job, provide the users with a Job Run Request form with the appropriate information filled in, that they can submit to Operations whenever they want the job run.

The above processes were taken from sections of the **“CYFD Information Technology Services Operations Process Document – Operations Guide”**. The Operations Guide provides ITS employees with a centralized source of information about processes, practices and change
management within the Information Technology Services service area. It provides operation
guidance and instruction. The processes are arranged according to the organizational unit
responsible for each process. The Operation Section (OPS) is responsible for the maintenance of
the Operations Guide. Below is a graphic which depicts how a software development project is
managed.

Figure 9.1.8  Software Development Project Management

Up to this point, we have described, in exceptional detail, the overall current governance
process employed by ITS. Next we will introduce a framework for Service Oriented
Architecture (SOA) Governance. Since this is our first attempt at creating an SOA
Governance Framework, the topic will be covered in more broad strokes – with more
questions than answers. More details will emerge when we start working with this new
technology.
9.2 SOA Governance

“SOA cannot be successful without governance”.¹ “SOA governance is not an option, it is an imperative”.² And finally, Figure 9.2.1, “Why SOA Governance Matters”, depicts another point of view on why SOA Governance is important. Without it, the IT community will just develop ‘a bunch of services’ (ABOS) – services that are not integrated, not re-usable.

Figure 9.2.1 Why SOA Governance Matters

```
Without Governance

A bunch of services

With Governance

The Promise of SOA
```

SOA governance is a subset of IT governance that focuses on the lifecycle of services: initial design, deployment, runtime and eventual changes to services. IT organizations must consider which operation is best suited to deploy different services under varying conditions and be able to monitor that the services are in fact enhancing the business value.

¹ ² Paolo Malinverno, Gartner, Inc.
The above shift to align business needs with IT capabilities is driving new governance needs. This new driving force is an extension of the IT governance described in Section 9.1. The key element in SOA is Service – re-usable services. SOA governance, then, focuses on these services and their lifecycle. SOA governance defines:

- What has to be done?
- How is it done?
- Who has the authority to do it?
- How is it measured?

It also mitigates many of the business risks inherent in SOA adoption by establishing decision rights, guiding the definition of appropriate services, managing assets and measuring effectiveness.
As the Constituent Services/Social Services Agencies start ‘walking’ down the path of SOA, they will need to clearly define roles, responsibilities and decision rights. Some representative questions that will have to be answered include:

- Who owns the data and is there agreement to allow the service access to the data?
- Who should fund the shared service? Who owns it?
- Who is responsible to fix it if it breaks?
- How is the business going to motivate the separate agencies to reuse enterprise assets and shared business services?
- Who makes a decision on whether a service can be accessible to other applications?

As the Agencies start to define services, the following type of questions will need to be addressed:

- What are the potential applications (service consumers) that will reuse this service?
- What are the common business services that are needed?
- What services already exist and are candidates for reuse?
- Which services can be shared and under what roles and circumstances?
- Which policies and standards can be factored as common across the geographical regulatory jurisdictions?
- Which are separate?
- Can the differences be isolated to maximize consistency while still being compliant?

As the Agencies begin to manage assets, the following type of questions will arise:

- How do we organize the shared business services and other enterprise assets so they can be effectively reused at a later date?
- Who is allowed to change a service reused by others?
- Who is using a given service?
- Who needs to approve the change?
- Who will be responsible for funding changes if we need to upgrade the infrastructure to meet specific performance or availability requirements at a later date?

The participating Agencies must agree on:

- Performance goals
- IT architecture standards for performance metric gathering and monitoring capability
- Service level agreements
Governance maximizes the value of SOA in aligning business needs with IT direction. Activities needed to establish, maintain and enhance an effective SOA governance framework are depicted in Figure 9.2.3 as a life cycle consisting of four phases: plan, define, enable and measure. This section provides information garnered from IBM whitepapers, websites, articles provided by IBM marketing representatives. They have done an excellent job in framing the issues in SOA Governance.

**Figure 9.2.3 The Four Phases of building a SOA governance framework**

SOA governance feeds from the SOA lifecycle: a sequence of tasks that starts with the Modeling of business services:

- Gathering requirements
- Simulating the flow of services
- Design of services

Next services are Assembled:

- Discovering who produced a specific service, what it does …
Constructing and testing the service
• Composing the set of services into a business process

Services are next Deployed:
• Integrated with people
• Integrated with processes
• Manage and integrate information

The final phase of the SOA lifecycle involves Management of services:
• Manage applications and services
• Manage identity and compliance
• Monitor business metrics

The foundation of the SOA lifecycle, however, is built on governance and best practices starting with the development of a Plan.

9.2.1 Plan

The planning phase of building a SOA governance framework focuses on understanding the overall scope of the governance opportunity within the organization and identifying areas for improvement. This phase includes:

• Committing to a strategy for SOA in the context of the overall business goals and IT strategy
• Explicitly determining the level of IT and SOA capabilities
• Articulating and refining the vision and strategy for SOA
• Reviewing current governance capabilities and arrangements

Most of these activities are people-centric and involve extensive collaboration. The required interaction can be simplified with sophisticated business modeling tools, as well as collaboration tools such as instant-messaging, e-mail, Wiki, dashboards, calendaring and role-based portals.

9.2.2 Define

Once the opportunities for improved governance are identified, business and IT people can work together to define and modify the current governance arrangements and mechanisms. New approaches to creating policies should be agreed on at this time. Other important governance decisions and mechanisms created during this phase may include:

• Establishing or refining an SOA Center of Excellence
- Defining any additional capabilities required, such as upgrades to the IT infrastructure
- Conducting staff training on an ongoing basis
- Agreeing on policies for service reuse across lines of business
- Putting funding mechanisms in place for encouraging this reuse
- Establishing mechanisms to guarantee service levels

These mechanisms and SOA governance decisions can speed the process of translating business design into IT design during the assembly phase of an SOA project.

### 9.2.3 Enable

Solutions to governance needs are put into action during this phase of establishing the SOA governance framework. These solutions may include deployment of new or enhanced governance arrangements. It is likely that communication mechanisms and education mechanisms will be rolled out to entrench the new governance arrangements within both the business and the IT decision-making communities. Governance activities within this phase influence how SOAs are deployed enabling the policy enforcement infrastructure.

### 9.2.4 Measure

During this phase, governance arrangements and mechanisms that were identified in the Define phase and deployed in the Enable phase are monitored. Activities occurring in this phase help ensure that the goals of the new governance framework are in fact being realized. If not, there is an opportunity for the business to refine and enhance its governance effectiveness by initiating a new cycle to enhance the SOA governance framework.

This section highlights the contents of an excellent white paper developed by webMethods: “SOA Governance – Enabling Sustainable Success with SOA” dated October 2006. Continuing with the theme that the SOA Governance section began with, “… Without SOA effective governance, organizations will have exceptional challenges which include:

- A fragile SOA implementation
- Services that cannot be re-used by developers because they were not designed for reuse or there is no easy method to discover what services currently exist
- Because there is no formal structure documenting services, developers will not have confidence in the integrity of the services
that exist. They will duplicate services which fundamentally defeats one of the core principles of SOA, i.e. reuse.

- Security breaches that cannot be easily traced
- Performance that is unpredictable and uneven

Given the above, the SOA initiative will be deemed a failure and will undoubtedly be discarded.

If we are to succeed with SOA and recognize the importance of governance to that success, what are some key steps that are required to get an organization on the right track?

The first imperative is architecture governance, a process that incorporates the following:

- Establishing enterprise technology standards
- Defining the high level SOA architecture and topology, as well as the infrastructure capabilities that the SOA should address
- Determining the SOA platform strategy and deciding what vendor products will be used
- Specifying the management, operations, and quality-of-service (QOS) (security, reliability, and availability) – characteristics of the SOA
- Establishing criteria for project design reviews

The final step in architecture governance is producing a roadmap that will guide the evolution of the architecture over time.

The key element in SOA is, of course, the services. SOA governance is inextricably intertwined with the lifecycle of those services. Thus, we have design-time, run-time and ‘change-time’ governance.

### 9.2.5 Design-Time Governance

Design-time governance is primarily an IT development function, involving the definition and creation of services. Policies that may apply include the process of validating that services are technically correct and valid – that they conform to organizational and industry standards. This would include the process of checking if services are compliant with Web Services Interoperability (WS-I) profiles. These checks can be automated by a tool called a registry. There can be alerts to advise that approvals and notifications are required by/to the appropriate personnel in the SOA chain. – approvals which attest to the architectural soundness of the services and the fact that they meet organizational standards.

Design-time governance will be of interest to the business analysts, architects and developers building the services. Key issues that one should consider include:
Determining the fit of a service as an enterprise asset
Determining which services should be built given the specific business requirements
Reviewing the design of services and validating that they conform to enterprise standards and practices
Determining how external services will be governed versus internally developed services

Finally, only authorized users will be able to publish, search and view service assets in the registry.

9.2.6 Run-Time Governance

The people, that this section would have special interest, are our IT Operations staff. Run-time governance specifically covers policies for controlling the deployment, utilization and operation of deployed business services. Key issues that one should consider include:

- Verifying that a service conforms to a set of rules before it is placed in production. For example, a service may require a specific message transport or certain schemas.
- Verifying that security is enforced. Only authorized personnel with the correct permissions are able to access the services. In addition, if it is necessary to encrypt data, verify that it is, indeed, encrypted.
- Verifying that corporate standards are adhered to for the specific business services

IT Operations will also be responsible for monitoring the execution of services and measuring performances against service level agreements (SLA) that are setup between IT and the consumers of the services. If it is determined that the SLA’s are not being met, the appropriate remedial actions will then be put in place.

The third type of governance that we will discuss is change-time governance.

9.2.7 Change-Time Governance

As services go through their normal lifecycle, some may require enhancements, modifications or simply depreciation (retired from use). Management of this process is significantly more challenging than design-time governance. There are more ‘moving parts’. With the SOA paradigm, technical personnel are not only exposed to the coding/deployment/customization phase of software/services development but also composition. Composition does not entail new code but rather orchestration of already existing business services into totally new business processes. There is also an
emphasis on making continual changes to metadata configurations. These include SLAs, new policies, contracts and constantly evolving processes. Change-time governance includes:

- Understanding inter-service relationships and dependencies
- Understanding what will be the impact of changing services in a run-time/production environment
- Understanding and managing the rollout of these services
- Understanding and managing changes in SLA’s and policies

Like design-time and run-time governances, the management of change-time governance can be facilitated by SOA infrastructure tools.

9.2.8 SOA Governance Technologies

SOA governance at its foundation is the ability of information technology and business users to enforce and automate policies across the business service lifecycle from design-time to run-time culminating in change-time. What are the key requirements or components that would comprise such an automated system?:

- A registry, which represents the ‘system of record’. It is a central catalog which documents the what, why, who of business services. It is the first place that a consumer of services would visit to see what services producers have created.
- A repository, which in many implementations is a component of the registry, which stores metadata on many of the business services' artifacts: SLA’s, policies, contracts
- Agents, which assist in the enforcement of policy and control design-time, run-time and change-time governances
- A rules engine, which manages the declaration of policies and rules and automating their enforcement

The final key ingredient in SOA governance is the user environment. It includes:

- Publishing of a service by an authorized provider
- Discovery of a service by a potential consumer
- Requesting the use of the service by that consumer
- Agreeing on the terms of service delivery
- Authorizing the consumer
- Provisioning of the service
- Monitoring of the service delivery.
In addition to the above, approval processes need to be implemented so that human beings are actively involved in the process and are not dependent on automation to their implementation peril.

9.2.9 Getting Started With SOA Governance

At what point in the SOA implementation process should one consider SOA governance? SOA governance should be an integral part of the overall SOA strategy. As such, the "big picture" should be addressed first:

- A SOA governance strategy must be a mandatory component of the broader SOA strategy. Therefore, governance should be in place with the introduction of a pilot project. This will not only prove the approach to SOA, but also the related governance practices along the way.
- Establish a governance roadmap. This will include specific governance capabilities that the organization wants to put in place, and how they will be implemented.
- Start at the beginning. Develop governance policies for the SOA architecture and then address design-time, next run-time and finally change-time governances.
- Put a governance system in place. You can not purchase a governance solution in a box. Creating the right technology framework will make the task easier.
10.1 Computing Environment Defined

A computing environment is any location where computer equipment is in use and the main users of the equipment have responsibilities for managing those systems.

In the Data Center, environmental controls, physical security, and fire control systems are obvious requirements for ensuring the facility is protected. In a large data center environment, personnel are normally asked with the management of these items, and periodic reviews are performed to determine the adequacy of the controls in place. In a smaller agency, county or work group computing environments, the computer systems are often regarded as office equipment and are not protected as closely as large scale ‘blade server’ or enterprise server systems. Computer systems, whether a stand-alone PC serving one user, or midrange processors connected to a LAN or WAN, supporting ten to twenty or even one hundred users, require additional attention when it comes to planning for a disaster situation.
10.2 Disaster Defined
A disaster (most people would agree), is any time the computer system relied upon to perform job functions is not available. A virus that destroys data in memory and causes a workstation to crash, a hard drive failure which results in data loss, a power outage, or a fire that destroys the office and your computer equipment is a form of a disaster. Problems of a lesser degree such as program errors, and temporary loss of network connectivity can also be considered disasters. The following topics for consideration will help the Department and its regional/county offices be better prepared for recovering from such occurrences in a timely fashion and can potentially reduce the likelihood of less serious problems becoming major catastrophes.

10.3 Planning for a Disaster
Disaster recovery planning involves the same actions for an agency or bureau computing environment as it does for a large scale ‘blade server’ data center.

The three basic steps to take are:

- Minimize Risk: Identify critical resources that must be protected, determine what threats those resources are vulnerable to, and develop methods to eliminate or reduce the vulnerability to those threats,
- Prepare for Disaster: Identify resources required to restore operations following a disaster, such as software, backup copies of data, and documentation, and
- Develop Recovery Plans: Design procedures to restore business operations following a disaster.

10.3.1 Minimize Risk
Efforts to minimize risk must be a continuing endeavor. Items such as inadequate insurance coverage, uncharged fire extinguishers, shared user ids with passwords that never change, untrained personnel, and open access to areas with mission critical equipment are all risks to any computing environment. As a function of building or office management these concerns must be addressed. Managing these risks for a computing environment becomes even more important when you consider the impact associated with the loss of equipment, data, and the impairment of performing business functions that rely on computerized applications.

10.3.1.1 Special Concerns for Computing Environments
A computing environment is a critical resource for doing business. If the environment is unstable or becomes unavailable, the ability to complete important functions is compromised. When tasks depend on large amounts of electronically stored information, performing them manually can often be an insurmountable task.

The computing environment consists of more than normal office space (furniture, telephones, copiers, fax machines, etc.). It is a complex configuration of the resources required to perform a specific function. The configuration is made up of a specific model.
of computer and peripheral devices, software, and communications. It may have taken
days, weeks, or even months to develop the required environment and reconstructing it
in short period of time can be extremely difficult, if in fact it can be rebuilt at all.

10.3.1.2 How to Protect Computing Environments

There are many levels of protection that can be implemented, however they should be
cost effective. For example, it does not make any sense to spend $50,000 to have
protection from a $5,000 threat. The following subjects offer data that can be used to
determine the level of protection that is appropriate for the departmental and bureau
computing environments.

10.3.1.3 Power Protection

Computer equipment should, at a minimum, be shielded from power surges. Unexpected
spikes in electrical current can cause system or component failures. Power
outages can also have a major impact on processing schedules and may cause data to
be lost. Depending on the criticality of the equipment and the processes that it supports,
some options for protection include:

- Surge protection to shield equipment from voltage spikes,
- UPS (Uninterruptible Power Supply) to protect from system crashes following a
  power outage. This allows for a graceful shutdown of systems (5 to 10 minutes),
  and
- UPS with backup power provided by a generator for extended conditioned power
  protection.

10.3.1.4 Fire Protection

Protection from fire damage due to smoking has been accomplished through a no
smoking policy in the State buildings. Some other relatively simple methods of providing
additional fire protection include:

- Installing fire detection and suppression systems,
- Smoke detectors, and
- Acquiring fire extinguishers for office areas.

For larger more critical computing environments a high level of fire protection should be
considered. The following controls should be in place:

- Sprinkler Systems (dry-pipe with electrical shut off)
- Automatic fire suppression (with electrical shut off)
10.3.1.5 Water Protection

Water or excessive moisture in a computing environment has the potential to cause electrical damage, and destroy paper records and magnetic media. Measures for protecting the environment from water damage include:

- Installing a drain system near potential hazard areas such as A/C units to control water leakage and condensation,
- Installing a monitoring and alarm system for A/C unit water overflow in computer rooms (where applicable), and
- Positioning computer equipment on upper levels of buildings if that building is located within a flood zone.

10.3.1.6 Physical Security

The most effective means of protecting computer systems is physical security. Physical security may be difficult to implement due to the nature of business conducted in the (300 San Mateo Building). Most of the building space in use by the Department was not originally designed to house computers, nor was it built with tight security in mind. Public access to most floors of the 300 San Mateo Building is necessary given the location of departments with varying security requirements.

For Desktop computers and LAN systems, physical security should consist of at least the following:
- Locks on entrances to office areas controlled by combination or key

For departmental or bureau computing environments that contain expensive equipment, critical and/or sensitive data, physical security should consist of at least the following:
- Locks on entrances controlled through magnetic card keys.

Stricter physical security can be achieved through the use of methods including:

- Installation of localized alarm systems to sound when unauthorized access occurs.
- Motion detection equipment to notify building security of intrusions.

10.3.1.7 Security Procedures

Security procedures for all computing environments should (and in some cases do) include:

- After hours monitoring by security,
- Periodic inspection when facilities are not in use,
- Keys and combination locks which are changed following employee terminations,
- Equipment inventory labeling, and
- Control forms and/or procedures for moving or relocating equipment.

10.3.1.8 Insurance

Regardless of the level of effort applied to minimize risk, it is impossible to eliminate all risks for any organization. Insurance coverage is therefore an important piece of the overall protection of the computing environments. Coverage should at least cover the cost of equipment replacement for the 300 San Mateo Data Center, as well as a minimal inventory of desktop computers. Since PCs and minicomputers depreciate rather quickly, replacement coverage is preferred over market value coverage. Other expenses to consider for insurance coverage include relocation and cleanup.

If the organization is self-insured, good documentation of losses will be essential in determining the cost of recovery. Complete inventories of equipment and software should be maintained and updated whenever equipment is purchased or changes to software levels are made. Annual physical inventories should be performed and compared to insurance coverage levels to assure that the required funds are available if needed.

10.3.1.9 Evacuation Plans

The most critical resource to protect is the people that support the organization. Facility evacuation plans should be in place for the entire building as well as departmental or work group areas. Evacuation plans should include the best escape route, a designated meeting area to account for personnel, and a list of the tasks (shutting down computers, gathering documentation, etc.) to perform.

10.3.1.10 Equipment and Software Inventory

Up-to-date lists of equipment and software including details such as manufacturer, model and/or version numbers, option settings, maintenance contracts, lease information, and dates of purchase are needed for:
- Ordering replacement equipment after a disaster or theft,
- Insurance claims,
- Coordinating upgrades,
- Identifying alternative processing site requirements,
- Re-ordering documentation for software products following a disaster, and
- Restoring computing environments following a disaster.
10.3.1.11 Software and Data Backup

A complete backup program must be in place for each computing environment. There should be at least one set (preferably two) of backup materials that can facilitate recovery of the systems. Duplex copies or the original version of backups should be stored at an off-site facility to enable the organization to recover from problematic situations including:

- File and/or data base recovery following a program abend or other outage which caused data loss,
- Operating system recovery needed as a result of hardware failure, and
- Complete system recovery needed as a result of a disaster which destroyed the entire office and/or equipment.

10.3.1.12 Off-Site Backup Storage

Backup copies of software, data, and documentation, including inventory lists, operational procedures, disaster recovery plans, organizational charts, hardware configuration diagrams, emergency contact lists, and escalation procedures should be kept at an alternate off-site facility to ensure their availability during an emergency situation. It is also a good idea to maintain an inventory report of items stored off-site for use during emergencies.

10.3.1.13 Virus Protection

As the use of networked personal computers increases, so does the likelihood of a virus attack. With Internet connectivity the risk of infection by virus is even greater. While it is impossible to completely protect all computers from viruses, there are actions that can be done to prevent them from entering the ITS computing environment. The basics are prevention, detection, and inoculation.

10.3.1.14 Prevention

Virus prevention can be managed by steps including:

- Avoid extensive downloading and execution of programs from bulletin boards or the Internet, and
- Minimize data sharing/moving via diskettes brought in from outside the department.

10.3.1.15 Detection and Inoculation

Virus detection and inoculation can be accomplished by the following:

- Installation of commercial virus detection and correction software, on Servers and stand alone PCs to scan hard drives and memory during login or startup.
Keeping current backups of software to facilitate recovery to a ‘clean’ version following a virus attack.

10.3.2 Preparing for Disaster

In order to prepare for a disaster it is imperative to analyze business functions that have to be performed, determine what resources are required to perform those functions, and to protect those resources. Absolute protection of all resources is very difficult. (for example, having locked doors does not prevent a break-in or theft). Protection of resources can be defined as determining how to have resources available following a disaster. A good example is ‘protecting’ data by backing it up and storing off-site copies that can later be used to recreate the information that had been destroyed.

10.3.2.1 Identification of Recovery Resources

Key resources differ according to the complexity of the computing environment, the number of users, and the technological experience and size of support staff. Performing a risk assessment to identify critical automated business functions is the first step in determining what resources will be necessary for disaster recovery. Once computer systems and applications have been prioritized, then identification of recovery resources can be focused on very critical areas.

Recovery information must be complete to the extent that it provides adequate data to reconstruct the computing environment following a disaster. Previous discussions about risk identified items that should be stored off-site. This information included such things as backups of software and data, organizational charts, hardware configuration diagrams, emergency contact lists, alternate site recovery instructions, and other documentation. Maintaining duplex copies at an off-site facility best protects these resources.

Developing an alternate processing strategy and having alternate resources available determines how quickly recovery can be made following a disaster. In the event that the computing environment consists of special equipment, or is the sole location with this type of equipment, getting protection means finding an alternate or backup processing site. Under certain disastrous circumstances, alternative departmental computing environments can be set up using floor space of another office building or alternate floor of the current building. In the case of the Data Center, which requires special environmental controls and raised floor, a hot-site should be considered as protection. Generally speaking, displaced workers can be relocated to the training or conference rooms for small-scale problems. A large scale building disaster will require the availability of extensive floor space and office equipment to restore operations.
10.3.2.2 Ensuring Off-Site Resources are Usable

The owners or managers of the computing environments should be ultimately responsible for making sure items needed for recovery are stored off-site. This should be done in conjunction with an overall disaster recovery plan that considers the organization of resources being stored off-site, and the delivery logistics. It is important to keep inventory reports and checklists for verifying which information is the most current. For example, if weekly data backups are sent off-site and kept there for four weeks, depending on the amount of data, determining which backups are current can cause serious delays in the restoration process. Additionally, reports and lists should be kept as short and simple as possible for ease of use. (500 page tape reports are not the easiest way to determine tape volumes required for restoring data).

Verification of resources being stored off-site should be performed periodically to ensure updates to procedures and documentation are applied. Furthermore, if the scope of recovery plans change, additional resources required for recovery must be added to the off-site inventory. Periodic quantitative analysis must also be performed to make sure that all of the expected resources are actually stored off-site.

Employees with responsibilities for resources that have copies stored off-site should be familiar with the off-site rotation procedures. Keeping associates aware of the significance of those resources is a very effective motivational factor in guaranteeing that the resources are maintained efficiently.

Disaster recovery planning meetings should be held periodically to inform new employees of the procedures and to review any changes to the operating environment that might warrant changes to off-site resources. Participation in regularly scheduled disaster recovery tests is a worthwhile educational experience for personnel to see how recovery resources are used and why they are needed.

10.3.2.4 Ongoing Disaster Preparedness

In order to maintain a high level of preparedness for reacting to a disaster situation, management has to include consideration for recovery in daily operations. Ongoing preparedness is facilitated by continuous updating of off-site resources, regularly scheduled testing of recovery plans, and disaster awareness on the part of all employees.

10.3.3 Developing Recovery Plans

An effective recovery plan must concentrate on assembling the necessary staff and all available recovery resources at the best available location to begin restoration of operations. Another critical element of a recovery plan is how to return normal operations to the original home site.
10.3.3.1 Restoring Operations

Following a disaster the management team must respond by establishing a control center for communications and coordination. Staff should be assigned to perform the following tasks:

- Retrieve recovery resources from the off-site location,
- Acquire replacement equipment and any other required resources, and
- Restore operations at an alternate site.

Until operations have successfully been established at an alternate site, the management team should concentrate on:

- Establishing a communications center for delivery of the FACTS to associates and the Public. Rumor control can be very helpful in minimizing the negative impacts of a disaster,
- Monitoring operations determine any changes to policy that are acceptable while in recovery mode,
- Ensuring the normal operations are continued. (backups taken and sent off-site etc.), and
- Planning to return operations to the home site as soon as possible.

All of the business units within the enterprise should have Business Continuity Plans. Recovery of the data centers and other computing resources relied upon for performing critical business functions is just one of the critical elements of a successful disaster recovery plan. Computer systems recovery only addresses such issues as planning for reconstruction of lost data, recovery of available data, and restoration of computer systems needed to support essential business functions.

The enterprise must also review the vulnerability of its physical facilities to a disaster. The organization should consider many other critical aspects of business continuity planning as well. Outside the issues and concerns for recovering computer systems, administrative and organizational areas must have plans for surviving a disaster. There are issues such as evacuation planning, building security policies, arranging for alternate office space, purchasing special office equipment, and developing emergency response teams that must be considered.
11 Environment Models

The purpose of the environment models is to graphically describe the server farm required to support the development and implementation phases of the project. They show the number of pieces of equipment needed and how they will all reside on the same Local Area Network segment. Each environment serves a specific purpose during the development and implementation phase. The Production Environment supports the deployed application. The layout of equipment in Figure 11.1-1: FACTS Production Environment Model represents a centralized view of this environment. The decentralized/distributed layout would differ significantly. The User Test, Training and Development Environments will all be similar centralized layouts of equipment. Development will support system development activities. User Test will support the system test and acceptance test activities. Training will support a centralized approach to training Program Staff. If the state wishes to utilize training centers located throughout the state then the layout of equipment may need to change. This is dependent on deployment/rollout planning prepared as part of the Implementation Phase project plan.

You will note that ‘blade server’ connectivity is planned for each environment. To support this, test environments (see Figure 11.2-1: FACTS Test Environment Model) may need to be created in the ‘blade server’ systems to allow for business scenarios to be developed to test updating those interfaces. It would not be recommended that testing, training and development of direct or indirect interfaces be accomplished against production data stored on the ‘blade server’. If these type of environments do not exist on the ‘blade server’ they will need to be created.

Only the production environment is prepared for the redundancy required for load balancing and reliability. If there is a desire to test these capabilities in advance of deployment, the User Test environment should be constructed similar to the production environment to allow for this testing to occur.
11.1 FACTS Production Environment

Figure 11.1-1: FACTS Production Environment Model
11.2 FACTS System Test Environment

Figure 11.2-1: FACTS System Test Environment Model

11.3 FACTS User Test Environment
Same as System Test Environment.

11.4 FACTS User Training Environment
Same as System Test Environment.

11.5 FACTS Development Environment
Same as System Test Environment.
### 12 Appendices

#### 12.1 Glossary

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACS</td>
<td>Affiliated Computer Services</td>
</tr>
<tr>
<td>ADE</td>
<td>Application Development Environment</td>
</tr>
<tr>
<td>AFCARS</td>
<td>Adoption and Foster Care Analysis and Reporting System</td>
</tr>
<tr>
<td>AFDC</td>
<td>Aid to Families with Dependent Children</td>
</tr>
<tr>
<td>APD</td>
<td>Advanced Planning Document</td>
</tr>
<tr>
<td>APS</td>
<td>Adult Protective Services</td>
</tr>
<tr>
<td>BLOB</td>
<td>Binary Large Object</td>
</tr>
<tr>
<td>CA</td>
<td>Certificate Authority or Computer Associates</td>
</tr>
<tr>
<td>CDMA</td>
<td>Code Division Multiple Access</td>
</tr>
<tr>
<td>Certificate Authority</td>
<td>A trusted internal body or trusted third party that vouches for an individual or device’s identity by issuing a certificate and its related private key.</td>
</tr>
<tr>
<td>CICS</td>
<td>Customer Information Control System</td>
</tr>
<tr>
<td>CLIENT</td>
<td>Central Repository for ALL Client Data</td>
</tr>
<tr>
<td>CORBA</td>
<td>Common Object Request Broker Architecture is a standard that defines for JAVA based implementations how objects are initialized so they can be used by the application software.</td>
</tr>
<tr>
<td>CPS</td>
<td>Children Protective Services</td>
</tr>
<tr>
<td>CSD</td>
<td>Conceptual System Design Document delivered during Phase 1 of this project.</td>
</tr>
<tr>
<td>CSED</td>
<td>Child Support Enforcement Division</td>
</tr>
<tr>
<td>Name</td>
<td>Definition</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets- gives Web-site developers and users more control over how pages are displayed.</td>
</tr>
<tr>
<td>DB</td>
<td>Database</td>
</tr>
<tr>
<td>DDE</td>
<td>Dynamic Data Exchange is a Microsoft Windows mechanism that allows separate programs executing on a single client to exchange data.</td>
</tr>
<tr>
<td>DHH</td>
<td>Department of Health and Hospitals</td>
</tr>
<tr>
<td>DHTML</td>
<td>Dynamic Hypertext Markup Language</td>
</tr>
<tr>
<td>Digital Certificate</td>
<td>A standard electronic file that declares the identity of an individual user, entity, device or application, and stores the certificate holder's public key.</td>
</tr>
<tr>
<td>DLL</td>
<td>Dynamic Links Libraries</td>
</tr>
<tr>
<td>DOC</td>
<td>Department of Corrections</td>
</tr>
<tr>
<td>DOL</td>
<td>Department of Labor</td>
</tr>
<tr>
<td>DP</td>
<td>Data Processing</td>
</tr>
<tr>
<td>DPS</td>
<td>Department of Public</td>
</tr>
<tr>
<td>DSD</td>
<td>Detailed System Design</td>
</tr>
<tr>
<td>ECMA</td>
<td>European Computer Manufacturers Association-dedicated to the worldwide standardization of information and communication systems.</td>
</tr>
<tr>
<td>EDI</td>
<td>Electronic Data Interchange- the transfer of data between different companies using networks</td>
</tr>
<tr>
<td>FACTS</td>
<td>CYFD Web Enabled Family Automated Client Tracking Systems</td>
</tr>
<tr>
<td>ERD</td>
<td>An Entity Relationship Diagram is a tool for designing databases that shows the kind and organization of the data to be stored in the database. It depicts the things of importance in an organization (entities), the properties of those things (attributes) and how they are related to one another (relationships). The ERD is independent of any data storage or access method.</td>
</tr>
<tr>
<td>Name</td>
<td>Definition</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Fat Client</td>
<td>An application architecture component where the majority of the executable code is contained on the client including the user interface, business logic and data access logic. Only the physical data access occurs on the database server.</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol- the protocol for exchanging files over the Internet/Intranet</td>
</tr>
<tr>
<td>GB</td>
<td>Gigabyte</td>
</tr>
<tr>
<td>GSD</td>
<td>General Services Department</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface- a program interface that takes advantage of the computers graphics capabilities to make the program easier to use.</td>
</tr>
<tr>
<td>HRS</td>
<td>Human Resources System</td>
</tr>
<tr>
<td>HSD</td>
<td>Human Services Department</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol - The underlying protocol used by the World Wide Web</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment - a programming environment integrated into a software application that provides a GUI builder, a text code editor, a compiler and/or interpreter and a debugger.</td>
</tr>
<tr>
<td>IS</td>
<td>Information Systems</td>
</tr>
<tr>
<td>ISV</td>
<td>Independent System Vendor</td>
</tr>
<tr>
<td>ITS</td>
<td>Information Technology Services Department</td>
</tr>
<tr>
<td>IVR</td>
<td>Interactive Voice Response</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol is a security implementation mechanism used to support user authentication, authorization and non-repudiation.</td>
</tr>
<tr>
<td>MB</td>
<td>Megabyte</td>
</tr>
<tr>
<td>Name</td>
<td>Definition</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MEDS</td>
<td>Medicaid Eligibility Data System</td>
</tr>
<tr>
<td>MMIS</td>
<td>Medicaid Management Information System</td>
</tr>
<tr>
<td>MOM</td>
<td>Message Oriented Middleware is a software package that will guarantee the delivery of a message to another transactional system when the message is received from a client. In some cases, data in the message may be manipulated as defined by rules processing.</td>
</tr>
<tr>
<td>NCANDS</td>
<td>National Child Abuse and Neglect Data System</td>
</tr>
<tr>
<td>n-TIER</td>
<td>An architecture characterized by an ultrathin (web) or thin client and several servers. (Several Application servers, Database server)</td>
</tr>
<tr>
<td>OLAP</td>
<td>Online Analytical Processing</td>
</tr>
<tr>
<td>OLE</td>
<td>Object Linking and Embedding</td>
</tr>
<tr>
<td>OTH</td>
<td>Other. (National Adoption Exchange, All CYFD Courts with child welfare jurisdiction</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure. The technology and processes that support the use of public key technology; these services may include key management, certificate issuance, certificate revocation, key and certificate validation, time stamping, message archiving, key recovery and digital notary services.</td>
</tr>
<tr>
<td>Plump Client</td>
<td>An application architecture component that is very similar to the ‘fat client’ but the data access logic has been moved to the database server.</td>
</tr>
<tr>
<td>POP</td>
<td>Points of Presence</td>
</tr>
<tr>
<td>QA</td>
<td>Quality Assurance</td>
</tr>
<tr>
<td>Name</td>
<td>Definition</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RDBMS</td>
<td>Relational Database Management System</td>
</tr>
<tr>
<td>RPC</td>
<td>Remote Procedure Call</td>
</tr>
<tr>
<td>RTA</td>
<td>Run Time Asynchronous is used when interfacing to a system using a MOM to request or update information that the application does not wait for the response.</td>
</tr>
<tr>
<td>RTS</td>
<td>Run Time Synchronous is used when interfacing to a system using a MOM to request or update information that the application waits for the response.</td>
</tr>
<tr>
<td>SACWIS</td>
<td>State Automated Child Welfare Information System</td>
</tr>
<tr>
<td>SAS</td>
<td>System Architecture Specification</td>
</tr>
<tr>
<td>SHARE</td>
<td>Statewide Human Resources, Accounting, and Management Reporting System.</td>
</tr>
<tr>
<td>SOA</td>
<td>Service-oriented Architecture – a software development methodology that is based on services; services that are abstract, autonomous, loosely coupled, composable, re-usable, discoverable and stateless based on standardized contracts.</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol - a lightweight XML-based messaging protocol used to encode the information in Web service request and response messages before sending them over a network</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>SRD</td>
<td>System Requirements Definition</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Socket Layer</td>
</tr>
<tr>
<td>TANF</td>
<td>Temporary Aid to Needy Families</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Transport Control Protocol/Internet Protocol</td>
</tr>
<tr>
<td>Thin Client</td>
<td>An application architecture component which contains only the user interface executable code at the client and can interface with one or more servers where the business logic and data access can occur.</td>
</tr>
<tr>
<td>Name</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Traditional (3270)</td>
<td>An application architecture which utilizes only one platform for execution of the user interface, business logic and data access.</td>
</tr>
<tr>
<td>Ultra Thin Client</td>
<td>An application architecture component whose user interface processing has been separated between a web server and the client. In the ultra-thin client model, two additional categories also exist. The first contains applets or ActiveX controls. The second is purely HTML processing.</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language - a notation combining the individual design mechanisms developed by Booch, Rumbaugh and Jacobsen.</td>
</tr>
<tr>
<td>WAN</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>WAP</td>
<td>Wireless Application Protocol - a secure specification that allows users to access information instantly via handheld wireless devices such as mobile phones, pagers, two-way radios, smart phones and communications.</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language is a standard method for transmitting and defining data to a browser.</td>
</tr>
<tr>
<td>XSL</td>
<td>Extensible Style Language- a specification for separating style from content when creating HTML or XML pages.</td>
</tr>
</tbody>
</table>
12.2 Example of Topic Paper

CM19: Approvals

Release 3.5.00
Date Certified: November 24, 2004

Introduction

There are two different situations within FACTS when approvals are needed for a piece of work. Therefore, there are two different types of approvals: assignment approvals and authorization approvals. The CM19: Approvals design topic discusses authorization approvals.

In the case of an assignment approval, a piece of work needs approval by someone else. The worker creates an assignment to the person from whom approval is needed. Once that approval has been documented, that approving individual may then assign the work to a worker other than the person who requested the approval. An example of this is a Protective Services Report. The Intake worker enters the report using SM01a: Protective Service Report, documenting acceptance of the report on the Decision tab of the Protective Services Reports tab folder. That worker then assigns that Protective Services Report to an Intake supervisor for approval. That supervisor then makes an acceptance decision on the same Decision tab of the Protective Services Reports tab folder. If the report is accepted for investigation, the supervisor links to or creates a case for that report and assigns the case to the appropriate investigation unit. These Assignment approvals are handled by using the information documented on the individual business process topic and the assignment function of FACTS.

The second, and more common, type of approval is an authorization approval. In an authorization approval, the work remains the responsibility of the person who requested the approval. An example of this type of approval is a case plan. The Treatment Worker completes a case plan with regard to the case and documents his/her own approval. The worker then seeks supervisory approval with regards to the case plan. This type of approval is handled by the Approvals process documented in the CM19: Approvals design topic. Other examples of work handled by the Approvals process are assessments, out-of-home placements, provider licenses and payment requests.
Windows
This approvals process is coordinated between the CM19: Approvals topic and the individual business processes for which approval is required. The Approvals common application function (CAF) consists of the Approval button, four windows and the Approvals tab, which workers have as a part of their desktop.
Approvals within FACTS are window based. For each window within the system for which approval is required, an approval level is defined. In addition, each user of FACTS is assigned a job class. This job class also has a defined approval level. When a worker documents approvals in the system, the approvals process uses login information to derive their approval level and compare that with the level necessary for the window. This determines the status that is assigned to the work and whether or not to prompt the worker to go to a higher level of approval.

An Approval button is located on each window in the system that requires approval. This button is what initiates the approval process and presents a drop down selection list comprised of Approve, Not Approve, Reroute, Recall or View. By making a selection from the drop down list the worker is able to document approval or rejection of work. The worker is also able to view the approval history with regards to the work.
If Approve is selected, the system documents that worker's approval and the worker is prompted to route the work for further approval if further approval is required.
If Not Approve is selected the system designates the work as not approved. Each person on the approval chain is able to view the Not Approve designation on his or her own approval outliner.
If Reroute is selected, the worker is prompted to route the work for further approval via the Create Worker Assignment window.
If Recall is selected, this entry is documented in the approval chain. Future approval of the work needs to be restarted from the beginning. If a piece of work is changed at any time during the course of the approval process, a Recalled record is generated and displayed as a part of the approval history on the outliner. The worker is then prompted to restart the approval process.
The approval history for a piece of work, accessed through the view option of an Approval button, remains on the Approvals tab of each individual, under the Approvals in Progress icon, until a final approval designation has been made. Work that has received its final approval designation is removed from each individual's Approvals tab forty eight hours after the final approval.
Window Overview
Upon a worker's completion and approval of a piece of work, if that document requires additional approval, the worker is prompted with this pop up window that displays his/her immediate supervisor's name. If the worker selects "OK" the supervisor displayed is prompted to approve the work. If the worker selects "Other..." they can select another person, whose approval level is the same or higher than their supervisor, to approve the work.

Window Information
Processing:
- Clicking on the "Other..." button opens the Create Worker Assignment window
- If OK is selected, and the work has not previously been saved, three rows are written to the Approval table:
  1. the worker initial row with status = initial, action = initial, person id and person source id = worker,
  2. the worker approval row with status = pending, action = approval, person id and person source id = worker and,
  3. the supervisor receipt row with status = pending, action = approval, person id = supervisor and person source id = worker
- If OK is selected, and the work has previously been saved without an approval documented, two rows are written to the Approval table:
  1. the worker approval row with status = pending, action = approval, person id and person source id = worker and,
  2. the supervisor receipt row with status = pending, action = approval, person id = supervisor and person source id = worker

Scan Window:
Because this window is modal, Scan Window cannot be executed.

Window – Supervisor Approval

Window Overview
Upon a worker's completion and approval of a piece of work, if that document requires additional approval, the worker is prompted with this pop up window that displays his/her immediate supervisor's name. If the worker selects "OK" the supervisor displayed is prompted to approve the work. If the worker selects "Other..." they can select another person, whose approval level is the same or higher than their supervisor, to approve the work.
"Other..." they can select another person, whose approval level is the same or higher than their supervisor, to approve the work.

**Window Information**

**Processing:**
- Clicking on the "Other..." button opens the Create Worker Assignment window
- If OK is selected, and the work has not previously been saved, three rows are written to the Approval table:
  1. the worker initial row with status = initial, action = initial, person id and person source id = worker,
  2. the worker approval row with status = pending, action = approval, person id and person source id = worker and,
  3. the supervisor receipt row with status = pending, action = approval, person id = supervisor and person source id = worker
- If OK is selected, and the work has previously been saved without an approval documented, two rows are written to the Approval table:
  1. the worker approval row with status = pending, action = approval, person id and person source id = worker and,
  2. the supervisor receipt row with status = pending, action = approval, person id = supervisor and person source id = worker

**Scan Window:**
Because this window is modal, Scan Window cannot be executed.

**Window – Create Worker Assignment**
**Window Overview**

If the worker does not choose to route the work to the name entered on the Supervisor Approval window, clicking the "Other..." button directs the worker to the Create Worker Assignment window, which she/he can use to select a different CYFD employee. This window functions in a similar way to the assignment outliner and allows the worker to route the approval to the appropriate CYFD staff member for the next level of approval.

The worker expands the outliner to the appropriate staff member, highlight that person's name and then click on Assign to route for approval.

**Window Information**

**Processing:**

Create Worker Assignment is a modal window and therefore cannot be minimized.

**Scan Window:**

Because this window is modal, Scan Window cannot be executed.

**Tab – Approvals**

![Tab Overview](image)

**Tab Overview**

The approval tab of the main desktop is divided into two sections. The first is for work that requires action of the person whose approval tab is open. This section has the "person at desk" icon followed by the worker's name on the first line of the window. Second is the Approval in Progress section, which is designated by the approval in progress icon. No action is needed for items listed under the Approval in Progress section. These two sections comprise level one of the approvals tab.

When the user double clicks on the level one icon, the approvals outliner expands to level two (shown above). Level two displays the cases or providers for whom pieces of work need approval.
When the user double clicks on the level two icon, the approvals outliner expands to level three, which displays the particular item related to the case or provider for which approval is being requested (Assessment above). Level three can be expanded to view level four. Level four shows the status of the work that needs an approval decision. Each row in the status line displays a number indicating the piece of work, the worker's name, the date of any action taken, the overall status of the piece of work and the status as it relates to the worker. In addition, comments can be associated with any of the status rows. This is done by accessing the Comments window using the right mouse button. If comments have been documented the pencil icon appears next to the checkmark icon on the status row. Pieces of work appearing under the 'Approvals in Progress' icon can be recalled or rerouted by a worker. For work that has been created by a worker but the approval process has not begun, an entry appears under that worker's Desk icon with a status of Initial. This entry acts as an additional reminder to the worker of outstanding work for which the worker eventually needs to document approval. Once the user makes an approval decision, the piece of work is removed from the Desk section on the outliner, but is retained on the Approval in Progress section until 48 hours after a final approval decision has been made.

**Tab Information**

**Processing:**
- Double clicking on level four of the outliner opens up the piece of work, allowing the user to approve the work. Note: sometimes the piece of work must be navigated to find the correct approval button. For instance, Licensing Action approvals bring up the License window, and the Options->Additional Licensing Actions pop-up menu item must be invoked to bring up the Licensing Action window that contains the desired approval button.
- Clicking the right mouse button on level four of the outliner opens the Comments Window.

**Scan Window:**
From the Utilities menu on the menu bar, select Scan Window to get a chart of all data displayed in the window along with where the data resides (table and field) in the database. By clicking all tabs and radio buttons before selecting Scan Window, all data associated with this window appears. The data can then be printed and/or saved in any of several formats (an Excel spreadsheet, for example). The following screen image shows the results of Scan Window, followed by an icon which, when double-clicked, initiates the Excel spreadsheet.

Note: the information displayed pertains to the first tab of the window (Ticklers).
Window – Comments

Window Overview
This window allows all workers on the approval chain the ability to document and view comments about any individual record in the approval history. Comments can be documented and viewed by highlighting one of the rows on the lowest level of the approvals outliner and clicking the right mouse button. Each person who is a part of the approval chain can update or view the comments associated with this piece of work.

A pencil icon is displayed next to the approval row on the approvals outliner to signify that comments have been documented.

Window Information
Processing:
Comments are saved to the approval description field on the approval table.

Window – Approval History
Window Overview
This window is accessed via the View selection on the Approval button of any FACTS window for which approvals have been documented. This provides the historical view of each person in the approval chain, their status and the date on which the status was designated.
The approval history for a piece of work can be accessed for as long as the work is stored in FACTS.

Window Information
Processing:
Approval History is a modal window and therefore cannot be minimized.
Scan Window:
Because this window is modal, Scan Window cannot be executed.

Inventories
Automated Messages
None
Checklists
None
Ticklers
None
Text Templates
None
Triggers
None

Batch Programs
CM19 – Approval History
Summary: This program moves approval rows to approval history 2 days after final approval.
See Topic Paper – CM19
U:\FACTSBAT\ProdDocs\Cm\CM19.doc

Issues
None

Revision History
1. August 30, 1996
2. December 12, 1996 (Final)
3. July 24, 1997 (System Documentation)
4. Nov. 24, 2004 (Version 3.3)
### 12.3 Example of System Design Document

**System Design Document**

*For R3.5.00CR820*

#### SM04a, Maintain Case Document Revision History

<table>
<thead>
<tr>
<th>Document Version #</th>
<th>Revision Date</th>
<th>Summary of Changes</th>
<th>Responsible Person</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>11/03/05</td>
<td>Enable track changes</td>
<td>CM</td>
</tr>
</tbody>
</table>
2. Modifications
2.1. On-Line Modifications
2.1.1. Maintain Child Information
2.1.1.1. GUI Changes
For each window, display a screen shot of the modified window (be sure to include a caption or label identifying the name of the bitmap) and describe the changes. Updated by Developer/BA

[Screen snapshots, List out the new fields added]

New Fields:
None

Modified Fields:

Figure 1: The previously adopted field for Maintain Child Information

2.1.1.2. Object Modifications
List new or modified objects in the library for this topic. Updated by Developer/BA
New:
None
Existing:
d_sm04a_case_part_mastert d_sm04a_parents

2.1.1.3. Background Processing
Include a description of the background processing, especially any changes to the background processing. If this is a change Describe how the process works now, and then how it works after implementing the change. Updated by Developer/BA

The “Select if child was previously adopted:” field currently defaults to “No”. This is a potential error for AFCARS reporting, Foster Care data element #16 and must be changed to default to null. This field is required if the Maintain Child Information window is saved. If the window is opened but not saved, the field will not be required. The field will remain editable until the case is closed.

Thus, if the field has a value, a Worker has entered it.

2.1.1.3.1. Field database column mappings
For each field on the screen (window or pop-up), describe the table and field where the data is stored in the database.

**Table**: CASE_PART  
**Element**: FL_ADPT

### 2.1.1.3.2. Background Processing
Describe any processing that goes on behind the scenes. Updated by Developer/BA

None

### 2.2. Inventories

#### 2.2.1. Reference Data

##### 2.2.1.1. Drop Downs
For each drop down, describe the values that appear in the list when the user clicks the down arrow. Also, include the table and field reference where the data is stored and if there is a special sort sequence for the data. Updated by Developer/BA

**New Items**
None
**Modified Items**
The “Select if child was previously adopted:” field will no longer default to “No” but will be null until a value is chosen.

**Deleted Items**
None

##### 2.2.1.2. List Boxes
For each list box, describe all data that appears and the table and field reference. Updated by Developer/BA

No list boxes.

#### 2.2.2. Automated Messages
List the name of the new or modified Automated Message, who it is sent to, what the Subject will read and what the message will contain. Updated by Developer/BA

**New:**
None
**Existing:**
None

#### 2.2.3. Checklists
Describe the Category and Type of the Checklist and the Checklist Items. Updated by Developer/BA

**New:**
None
**Existing:**
None

#### 2.2.4. Ticklers
Describe the new or modified tickler. Updated by Developer/BA

**New:**
None
**Existing:**
None
2.2.5. Pop-up Message
For each pop-up message, list the ID_EXC_MESSAGE from the EXC_MESSAGE table (where these messages should be stored) and display a screen shot of the pop-up including a description of the text and buttons. (Once again, be sure to include a caption or label identifying the name of the bitmap.) Updated by Developer/BA

New:
None
Existing:
None

2.2.6. Text Processing
List new or modifications to existing text documents, including the Name of the document, the ID_REPORT and the TX_REPORT values. Updated by Developer/BA

New:
None
Existing:
None

2.2.7. Window_Usage
For any new or modified window, list the name of the window and 1) the number of tabs on the window, 2) the number of ticklers created/deleted from this window, 3) the number of window or system validations and edits, 4) the number of and complexity of associated text documents and 5) the number of related batch processes or extended online processes. These components will comprise a “Window Weight”. Updated by Developer/BA

New:
None
Existing:
None

1. Maintain Child Information
2. No tabs, Modal window
3. One validation (cannot select Participant for more than one of the following fields: Child’s Mom, Child’s Dad, Child’s Guardian (1) or Child’s Guardian (2))
4. No associated text documents
5. No related batch or online processes.

2.2.8. Triggers
(List the new or modified Trigger, including the CD_TXN, that will communicate from the online function to batch to initiate a batch process) Updated by Developer/BA

New:
None
Existing:
None

2.2.9. Batch Modifications
Describe any new or modified batch processes that are involved in fulfilling this request. Updated by Developer

No batch modifications

2.3. Database Modifications
2.3.1. Database Inventory
Any new or modified database tables must be coordinated with the DBAs in TSS. Note: be sure to fill out and attach 'Database Schema Change Form.doc'. List file names of sql queries or text documents. Updated by Developer

2.3.2. DSS/View Changes
New tables or changes to tables often affect DSS Views. All views that reference tables being changed must be reviewed to determine whether the table changes must be reflected in the corresponding views. List the new views or changes in this section. Updated by Developer/BA
3. Data Issues

3.1. Conversion Issues
Changes to data structures (tables, indices, etc.) as well as the data contained in tables may be updated via a “one shot” process. These must be organized and scheduled to be run during the implementation of the release. Describe these issues and describe the fix. Updated by Developer/BA

Conversion issues to be handled by CR821 by Batch team prior to v. 3.5.00 implementation.

3.2. Data Cleanup Issues
Cleanup of data involves correction of incorrect, obsolete or corrupt data, removal of test data, etc. If specific data sets must be cleaned up before this change is implemented, it must be described in detail here, and coordinated with the other implementation tasks. Describe these issues and describe the fix. Updated by Developer/BA

Cleanup issues to be handled by CR821 by Batch team prior to v. 3.5.00 implementation.

3.3. Security Changes Review (HIPAA)
All new functionality or changes must be reviewed for their impact on HIPAA. Any relationship to transactions and code sets, or to privacy processing must be detailed here along with a mitigation strategy. Updated by Developer/BA

No security changes affecting HIPAA.

3.4. Impact to APS with PS Change
New functionality or changes requested by PS or that have impact to PS must be evaluated with respect to the impact to Adult Protective Services, now administered by NM Aging & Long Term Service Department. Updated by Developer/BA

Although this window will be available to APS, the title will remain “Maintain Child Information” and it is not anticipated this window or the new questions on it will be used often by APS workers.

4. Documentation Updates

4.1. Topic Paper Updates:
(Enter name of the applicable Topic Paper. If the change being implemented does not require a change to any Topic Paper, enter "None" under Topic Paper Name) Updated by Developer/BA

Topic Paper Name: SM04a, Maintain Case

4.2. Data Dictionary Updates
Any additions or changes to the data dictionary; Table Descriptors, Code Groups, etc., must be detailed here. Updated by Developer/BA

4.3. Release Notes
- Throughout the System development phase, ADS and Divisions will create Release Notes. These notes will be distributed to the field at Release implementation. Initially the ADS Business Analysts will create the release notes; Divisions will revise/create their version of the release notes. If the Division Release Notes are created, these will be the Notes distributed to the field; else the ADS Release Notes will be distributed to the field. Updated by BA/Divisions

Business Analysts Release Notes:
PSD – Previously, the “Select if child was previously adopted:” field defaulted to “No” which could have caused an error incorrect information being reported on a federal reporting. Now, the field will not default to anything and a Worker must
choose a drop down value, if appropriate. This field is required if the Maintain Child Information window is saved. If the window is opened but not saved, the field will not be required. The field will remain editable until the case is closed.

Division Release Notes:
**Appendix A – Internal Review/Approval**

Notations:
- The Supervisor assigns each of the types of review to a team member by entering the team members name in the Assigned Person column.
- Reviews are conducted in the order listed in the table.
- As each person conducts the appropriate review he/she enters initials and the date in the appropriate columns.
- When the specific review is complete the next person on the list is notified.

### Document Version #: 1 PRE-DESIGN SESSIONS

<table>
<thead>
<tr>
<th>Type of Review</th>
<th>Assigned Person</th>
<th>Initials</th>
<th>Date Reviewed/Approved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Review*</td>
<td>Suresh Penumatsa</td>
<td>SP</td>
<td>9/13/05</td>
</tr>
<tr>
<td>Batch Review</td>
<td>Mary Bradford</td>
<td>MB</td>
<td>9/9/05</td>
</tr>
<tr>
<td>BA Review*</td>
<td>Patricia Ryan</td>
<td>PMR</td>
<td>09/09/2005</td>
</tr>
<tr>
<td>ADS Mgmt Review*</td>
<td>Ellen Davis</td>
<td>ED</td>
<td>09/09/2005</td>
</tr>
<tr>
<td>External IT Review</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*Required Review</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Document Version #: 2 DESIGN

<table>
<thead>
<tr>
<th>Type of Review</th>
<th>Assigned Person</th>
<th>Initials</th>
<th>Date Reviewed/Approved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Review*</td>
<td>Suresh Penumatsa</td>
<td>SP</td>
<td>11/10/05</td>
</tr>
<tr>
<td>Batch Review</td>
<td>Mary Bradford</td>
<td>MB</td>
<td>11/8/05</td>
</tr>
<tr>
<td>BA Review*</td>
<td>Patricia Ryan</td>
<td>PMR</td>
<td>11/10/2005</td>
</tr>
<tr>
<td>ADS Mgmt Review*</td>
<td>Ellen Davis</td>
<td>EED</td>
<td>11/18/05</td>
</tr>
<tr>
<td>External IT Review</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*Required Review</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Repeat the entire block for each version/revision.
Appendix B – Design Document Change Log

Notations:
. The intent of the Design Document Change Log is to document all changes made to the Change Request following the original Design Document creation. The owner of the CR will be responsible for logging all changes.
. Change Requests will only be accepted in e-mail format. Verbal or telephone requests to modify a Change Request are not acceptable.
. Change Log entries may not be reflected in the Design Document.
. This Change Log must always be included in the packet distributed to the User for review and approval.

<table>
<thead>
<tr>
<th>Change Item #</th>
<th>Date</th>
<th>Requestor: Division/Person</th>
<th>Batch Change?</th>
<th>Design Doc Change?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email Change Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Change Item #</th>
<th>Date</th>
<th>Requestor: Division/Person</th>
<th>Batch Change?</th>
<th>Design Doc Change?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email Change Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
12.4 References

AMS eCycle: System Development for eCommerce

Barry, Douglas K., - Web Services and Service-Oriented Architectures: The Savvy Manager's Guide (The Savvy Manager's Guide); April 1, 2003

Bieberstein, Norbert, Bose, Sanjay, Fiammante, Marc and Jones, Keith, Service-Oriented Architecture (SOA) Compass: Business Value, Planning, and Enterprise Roadmap (Developerworks); October 25, 2005

Brown, Paul C., Succeeding with SOA, Addison Wesley, April 2007.


Erl, Thomas, - Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services; Prentice Hall, April 16, 2004

Erl, Thomas, - Service-Oriented Architecture: Concepts, Technology, and Design; Prentice Hall, August 2, 2005


Jacobson, Ivar; Booch, Grady; Rumbaugh, James; *The Unified Software Development Process*, Addison-Wesley, Reading, 1999.

Jones, Steve – *Enterprise SOA Adoption Strategies*; November 13, 2006


Krafzig, Dirk, Banke, Karl, and Slama, Dirk – *Enterprise SOA: Service-Oriented Architecture Best Practices* (The Coad Series); November 9, 2004

Larson Editors, *Secrets of SOA An Enterprise View on Service-Oriented Architecture Deployment Revealed*; October 18, 2006

Marks, Eric A. and Bell, Michael - *Service-Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology* - April 28, 2006

McGovern, James, Sims, Oliver, Jain, Ashish, and Little, Mark – *Enterprise Service-Oriented Architectures: Concepts, Challenges, Recommendations* April 28, 2006


Pulier, Eric, Taylor, Hugh, and Gaffney, Paul – *Understanding Enterprise SOA*; November 1, 2005


Woods, Dan and Mattern, Thomas – *Enterprise SOA: Designing IT for Business Innovation*; April 28, 2006
### 12.5 Application Architecture Alternatives (Final)

In considering CYFD’s SACWIS System Application Architecture, specific evaluation criteria were developed to be applied to the various architecture alternatives. These criteria have been weighted among the alternatives to achieve an overall measurement of the user, business and data services as a basis for architecture selection. The following table compares the various architecture alternatives to each other based on the following criteria.

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Traditional 3270</th>
<th>Fat/Plump</th>
<th>Thin Ultra-thin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Required Functionality</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Online Performance and Response Time</td>
<td>4</td>
<td>3.5</td>
<td>3</td>
</tr>
<tr>
<td>Batch Processing</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Timely Reporting</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Ease of Use and Robustness of the GUI</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Ease of Interfacing to Other Systems</td>
<td>3</td>
<td>4</td>
<td>4.5</td>
</tr>
<tr>
<td>Ease of Deployment to private agencies</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Ease of Deployment to ITS Sites</td>
<td>4</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Integration with Desktop Applications</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Ease, Speed, and Cost of Development</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Security</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Compatibility with Existing IT Infrastructure</td>
<td>5</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Scalability as Workload Increases</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Reliability</td>
<td>4.5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Maintainability</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Manageability</td>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Cross-Platform Support</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Ease of User Support</td>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Maturity</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Evaluation Criteria</td>
<td>Traditional 3270</td>
<td>Fat/Plump</td>
<td>Thin Ultra-thin</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------------------</td>
<td>-----------</td>
<td>----------------</td>
</tr>
<tr>
<td>Required Hardware and Software</td>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Total Cost of Ownership</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Total</td>
<td>67.5</td>
<td>70.5</td>
<td>87.5</td>
</tr>
</tbody>
</table>

**Legend:**
0 - Not Applicable 1 - Lowest Rating 5 - Highest Rating
### 12.6 Single vs Multi-Vendor ADE

In considering CYFD’s SACWIS System Application Architecture, specific evaluation criteria were developed to be applied to determine whether a Single or Multi-Vendor approach would better serve the development of the system. These criteria have been weighed to achieve an overall measurement of user, business and data services as a basis for selection. The following table compares the two alternatives to each other based on the following criteria.

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Single</th>
<th>Multi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Required Functionality</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Online Performance and Response Time</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Batch Processing</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Timely Reporting</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Ease of Use and Robustness of the GUI</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Ease of Interfacing to Other Systems</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Ease of Deployment to private agencies</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Ease of Deployment to ITS Sites</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Integration with Desktop Applications</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Ease, Speed, and Cost of Development</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Security</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Compatibility with Existing IT Infrastructure</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Scalability as Workload Increases</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Reliability</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Maintainability</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Manageability</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Cross-Platform Support</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Ease of User Support</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Maturity</td>
<td>E</td>
<td>E</td>
</tr>
</tbody>
</table>
## Evaluation Criteria

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Single</th>
<th>Multi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required Hardware and Software</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>Total Cost of Ownership</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Standards</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Licensing</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Training</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Integration with Modeling Tools</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Integration with QA Testing Tools</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Integration with Version Control Tools</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Vendor Stability (added)</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Outsourcing Capability</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td><strong>Total (when not equal)</strong></td>
<td><strong>82</strong></td>
<td><strong>75</strong></td>
</tr>
</tbody>
</table>

**Legend:**

E - Even 1 - Lowest Rating 5 - Highest Rating
12.7 Thin Versus Ultra-Thin

In considering CYFD’s SACWIS System Application Architecture, specific evaluation criteria were developed to be applied to determine whether a Thin or Ultra Thin Architecture would better serve the development of the system. These criteria have been weighed to achieve an overall measurement of user, business and data services as a basis for selection.

<table>
<thead>
<tr>
<th>Evaluation Criteria</th>
<th>Thin</th>
<th>Ultra-Thin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support Required Functionality</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Online Performance and Response Time</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Batch Processing</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Timely Reporting</td>
<td>+1</td>
<td></td>
</tr>
<tr>
<td>Ease of Use and Robustness of the GUI</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Ease of Interfacing to Other Systems</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Ease of Deployment to private agencies</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Ease of Deployment to ITS Sites</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Integration with Desktop Applications</td>
<td>+1</td>
<td></td>
</tr>
<tr>
<td>Ease, Speed, and Cost of Development</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Compatibility with Existing IT Infrastructure</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>Scalability as Workload Increases</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Reliability</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Maintainability</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Manageability</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Cross-Platform Support</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Ease of User Support</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Maturity</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td>Evaluation Criteria</td>
<td>Thin</td>
<td>Ultra-Thin</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>------</td>
<td>------------</td>
</tr>
<tr>
<td>Required Hardware and Software</td>
<td>+1</td>
<td></td>
</tr>
<tr>
<td>Total Cost of Ownership</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>+3</td>
<td><strong>+9</strong></td>
</tr>
</tbody>
</table>

**Legend:**
0 - Both options are equal +1 - This option is favored
12.8 CYFD State Office Locations

CYFD Office Locations

Updated 2/16/07
12.9 CYFD Hardware/Network Topology

[Diagram of CYFD Hardware/Network Topology]

- CYFD Remote Frame-Relay To ATM Sites (IPVC)
- Qwest OC-3
- GSD/ISD
- Private OC-3 with L2 Circuits
- GSD ATM Switches In Bank Building
- ATM Switch
- Cisco 7204 Core Router
- Cisco 6506 CORE Switch
- Polycom

[Legend]
- Red: Untrusted Traffic
- Blue: Trusted Ethernet
- Orange: Untrusted Circuit
- Green: Trusted Circuit
- Yellow: Trusted and Untrusted Circuits

As of 1/1/06
12.10 Children Youth and Families Department Typical Remote Office

Logical Office Configuration

LEGEND
Red – Ethernet
Purple – ATM
12.11 Network Overview CYFD & State Agencies

5301 Central Ave NE & 300 San Mateo Blvd NE, Albuquerque

Internet

Bank of the West Building
5301 Central Ave NE
Albuquerque

Untrusted Public Network
OC-3

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Core Router
Cisco 7204

Core Switch
Cisco 6506

Firewall
Cisco 515e PIX

Video Conferencing Switch
Cisco 2940

Qwest Services Cloud

8 - Remote CYFD Video Conferencing Sites

53 - Remote CYFD Sites

ISD ATM Switch
Cisco 8510

ISD Router
Cisco 3640

ISD OC-3

Qwest Services Cloud

ISD Network
Simms Building
Santa Fe

Router

Switch

Firewall

15th Floor - OCIO

Computer
Printer
Switch
Router

10th Floor

Governor’s Office

Computer
Printer
Switch
Router

2nd Floor

Governor’s Office

Computer
Printer
Switch
Router

3rd Floor

Livestock Board

Computer
Printer
Switch
Router

9th Floor

Livestock Board

Computer
Printer
Switch
Router

Internet Router
Switch

Computer
Router
Switch
Printer

5301 Central Ave NE & 300 San Mateo Blvd NE, Albuquerque

Private Network
Fast Ethernet

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Video Conferencing Network
100FX

Private Network
Fast Ethernet

Video Conferencing Network
100FX
12.12 CYFD Database Infrastructure
CYSF Database Infrastructure

Sybase 12.5.3
Primary ASE 30GB
2.75GB RAM
12.0 Replication Server

HP-9000 sp7400
5CPU
5GB RAM
va7100
4K/150 GB Disk
SAN 1000GB Disk

HP-9000 sp7400
6CPU
7GB RAM
va7100
200 GB Disk

G:/TSS/Database/Reference_timeless/CYFDHOSTInfrastructure

Feb 19, 2007
### 12.13 Current Technical Environment Overview

**HARDWARE, SOFTWARE AND TELECOMMUNICATIONS EQUIPMENT** The SACWIS application must be designed to be flexible enough to work in the projected environment which will exist at implementation yet also operate in the ITS platform environment which currently consists of the following equipment and software:

| Existing PC’s:       | Gateway E2300  
|                      | Processor: Intel® Pentium® Processor 3.0Ghz 800FSB 1MB L2 Cache  
|                      | Memory: 512MB DDR Dual Channel 333/400MHz SDRAM (1 x 256 DIMM)(expandable to 2GB)  
|                      | Network Adapter: Integrated Marvell 10/100 Twisted Pair Ethernet  
|                      | Video: Integrated Intel® Direct AGP Graphics with up to 64MB Dynamic Video Memory Technology  

| Existing Field Application Dual Servers: | Dell  
|                                          | Hewlett Packard  

| Existing Communications Equipment:      | Switches – WS-C1924en,1912en, 2940-8tf-s,2948,6506-s1,2950-e-24-e1,3550-12g,3550-24fx-smi,3550-48emi,3508,3524 &3548-xl-en  
|                                          | Routers - Cisco 2620xm,7204,2610,2501 & 2524  

| Existing Wiring:           | Cat-5  

| Existing Operating Systems, Network Applications: | Novell Netware 6.0  
|                                                 | Zenworks for Desktops 6.0  

| ManageWise Client Components: | All Workstation have TCP/IP configured via DHCP  


<table>
<thead>
<tr>
<th>Operations Recap (monthly):</th>
<th>Batched Jobs Processed 36,000 Checks Printed 100,000 Total Cartridge Library 15,800 Xerox Printer Print Lines 52,360,000 Other(Laser) Printer Print Lines 110,000,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Environment:</td>
<td>HP 9000. The Operating System is HP-UX Version 11.11. The processor has 5 gigabytes of main memory and 5 CPUs</td>
</tr>
<tr>
<td>UNIX:</td>
<td>HPUX 11.11</td>
</tr>
<tr>
<td>Database:</td>
<td>SYBASE 12.5.x</td>
</tr>
<tr>
<td></td>
<td>SYBASE 15</td>
</tr>
<tr>
<td>Personal Computer/Networking Software:</td>
<td>Office 2000, Access 2000, Outlook 2003, Adobe Acrobat 5.0, Netware 6.0 …. See G:\Mgmt\Reporting\Inventory\Software Inventory.xls for a list of all software inventory.</td>
</tr>
</tbody>
</table>
12.14 ADE Criteria Issues/Questions Support Application Functionality

The following is a suggested list of questions that should be presented to vendors of ADEs:

- What are the vendors recommendations for building complex GUI’s?
- Does the vendors ADE provide access to additional GUI classes/libraries?
- Does the ADE provide pre-built access to legacy information?
- Does the ADE import the needed components?
- What are the tools for generating and managing documents?
- What tools are provided for constructing database queries and editing stored procedures?
- Are these additional pre-built components for application building?
- What web browser does the vendor support/recommend/require?

Ease of Learning and Use

- What is the typical learning curve for:
  - experience with a similar tool?
  - without experience in a similar tool?
- Are wizards provided for repetitive tasks? Can they be extended?
- Does the code editor include color coding, syntax checking, auto-completion and other aids? Are visual tools provided for assembling components? What views are provided for organizing the code (by component, project, package etc)?

Developer Skill Set Availability

- Is there a certification system for the ADE? If yes, what are the numbers for the US? For NM?
- Can the vendor provide a substantial list of organizations with staff experienced/certified in the ADE?

ADE Performance

- What is the recommended configuration of a developers individual environment?
- What is necessary beyond the individual environment? It's configuration?
- What tools are provided for profiling and performance tuning?
- What is the performance relative to other ADE’s?

Application Performance Developed using the ADE

- What is the largest client using the ADE?
- What is the largest application developed using the ADE?
- What are the specs for this application (#users, avg. Response time)
- How much overhead is added by the ADE to the developed code?
**Scalability**
What is the size of the largest team that used this ADE?
What can cause a noticeable performance degradation for this ADE?

**Ease of Integration with Other Tools**
Does it provide an API for integrating third-party tools?
What and how many add-on’s are available?

What libraries and components for the ADE are available (both vendor and third-party)?

Does it support open architecture for including outside tools (design, debug, test, etc.)? What tools/components are provided for integrating with middleware and/or legacy systems?
Can developers generate, import and edit IDL libraries?
Can developers browse an ORB’s interface repository?
What support does the ADE have for XML? Parser? Libraries? Components?

**Visual Tools**
What other tools does this ADE integrate with?
What other tools has this ADE been successfully integrated with?
Does the ADE integrate with industry leading HTML editors?

**Web Enabled Features**
How does the ADE support the development of N-tier, ultra thin applications?
What level of HTML and Web stands does it require?

**Security**
What security features are available thru the ADE?
Does the ADE support CDMA and PKI security?
Does the ADE support single Sign on?
Does the ADE support certificates?

**Follow Industry Standards**
Does the tool conform to recognized web application industry standards? Which standards?

**Vendor Stability**
What is the vendors market share in ADE’s?
What is the amount of revenue generated by this ADE?
How is this tool related to the vendor’s long-term direction?
What is the number of developers dedicated to this ADE?
What is the release schedule for this ADE?
Are drastic changes being done?
What platforms are supported first?

**Vendor Services and Support**
How significant and extensive are partnerships with complementary technology vendors? (Modeling, middleware, web servers etc.) What is the support structure and associated costs for this vendor? How is support handled?

**Documentation**
Does the documentation include tutorials?
Task oriented Users guide?
Context sensitive help?
All API's including standard library/component functions are fully documented?

**Maintainability**
Ability to Reverse engineer programmatic representation to a visual one?
Limitations? Does the ADE support a code analyzer?

**Supportability**
Does the ADE have distributed debugging facilities?
Which debugging API's does it support?

**Productivity**
Does it provide wizards? Can they be added to?
Does the visual tool allow for additional objects to be added to it's palette?
Is drag and drop supported of placing objects on the GUI?
Does the ADE allow for selecting/copying/changing attributes of multiple objects at the same time?
Can one visually wire all elements?
Ability to access all methods/objects?
Ability to edit existing interactions?
Ability to selectivity show interactions?
Does the tool provide query and transaction building wizards?

**Cross Platform Support**
What platforms does the tool run on?
Is there support to deploy the application to Unix like systems? (UNIX, Linux, OS/390, AS400)?

**Portability**
What versions of interfaces are supported required? (Java, DCOM etc.)
Can the interfaces be changed independently of the tool version?
Is AWT (abstract windowing toolkit), Foundation Classes supported?
On which underling library levels are the visual components based?
Is JIT (just in time) and native compiles supported on target platforms?

**Licensing**
What are the license options for the vendors ADE?
Training
What is the recommended training environment?
Number of Days?
Where is the training available?
Cost of the training?

Data Modeling
Does the ADE provide support for data modeling?
Which ones?

Process Modeling
Does the ADE provide support for Process modeling?
Which ones?

Object Modeling
Does the ADE provide support for object and component modeling with UML?
Which ones?

QA Testing
Does the ADE provide for both manual and automated regression testing?

Version Control/Configuration Management
Does the ADE have its own version control tools and repositories?
If so describe?
Does the ADE interface to third-party version control tools?
If so which ones?
What tools are provided to manage configurations and simplify deployment?

Total Cost of Ownership (TCO)
What are the support costs for the ADE?
What are the costs of the ADE and required additional products?
What are the training costs?
What are the yearly support costs?
Do new versions require new payments?

Application Server
What Application servers are supported?
What platforms?
Are their restrictions on the application server environment?

Database
What database standards are supported?
Are there performance implications of specific databases?
12.15 Confidentiality Building Block Cryptography

Of all security mechanisms, cryptography is the one most suited to open and hostile environments, where control is otherwise limited, environments like the modern, open, flat, broadcast, packet-switched, heterogeneous networks.

Cryptography is broadly applicable. In the presence of cheap computing power, its uses are limited only by our imaginations.

It is portable; the necessary software to encode or decode the information can be distributed at or near the time of use in the same package and channel.

It is effective and efficient; that is to say, it is usually the cheapest way to achieve a specified degree of protection.

It’s low cost is the result in part of the low cost of the modern computer, and it is falling with the cost of that computing.

Modern cryptography is arbitrarily strong, that is, it is strong as we need it to be. It is significantly stronger than other security mechanisms. Almost never will cryptography be the weak link in the chain, for example, key management. The cryptography component of a security solution is robust and resilient, and not likely to break.

One way of looking at cryptography is that it changes the problem of maintaining the secrecy of the message to one of maintaining the secrecy of the keys.
12.16 Non-Repudiation Building Block

Digital Signatures and Certificates

If identification and authorization makes use of digital signatures, then certificates are required. They can be issued by the organization or by a trusted third party. Commercial public key infrastructures (PKI) are emerging within the Internet community. Users can obtain certificates with various levels of assurance.

Level 1 certificates verify electronic mail addresses by the use of a personal information number that a user would supply when asked to register. Level 2 certificates verify a user’s name, address, social security number, and other information as in a credit bureau database. Level 3 certificates are available to big companies and government. This level of certificate provides photo identification with the future use of smart cards and other hardware token.

Many of the latest web servers and web browsers incorporate the use of digital certificates. Secure Socket Layer (SSL) is the technology used in most Web-based applications. SSL version 2.0 supports strong authentication of the Web server, while SSL 3.0 adds client-side authentication. Certificates at the user end should be used in conjunction with standard technologies such as SSL to provide continuous authentication to eliminate the risk of session hijacking. Access to digital certificates stored on personal computers should be protected by passwords or pass phrases. All policies for password management must be followed and enforced.

PKI and the Digital Certificate

The PKI is a system for publishing the public-key values used in public-key cryptography. There are two basic operations common to all PKI’s:

- Certification is the process of binding a public-key value to an individual organization or other entity, or even to some other piece of information such as a permission or credential.
- Validation is the process of verifying that a certificate is still valid.

A digital certificate is a credential (in digital form) in which the public key of the individual is embedded along with other identifying data. That credential is encrypted (signed) by a trusted third party or certificate authority (CA) who has established the identity of the key owner. By ‘signing’ the certificate, the CA establishes and takes liability for the authenticity of the public key contained in the certificate and the fact that it is bound to the named user.

Certificate Authorities

Who are CA’s? Some large institutions are their own CA’s, like banks (private CA’s). There are some independent services (public CA’s) developing, and government, using the licensing model as a take off point, moving into this environment. Public CA’s will become a new security industry.
The certificate is the heart of a PKI system. Any population of users who wish to intercommunicate selects or is required to use a specific CA to obtain a certificate. That certificate contains indicative information about the target individual. This individual is referred to as the ‘distinguished name’ - implying that there can be no ambiguities in certificate-based identification - all Smiths must be separately distinguished by ancillary data.

**Where are Certificates used?**
Certificates are used primarily in open environments in which closed network security techniques are inappropriate or insufficient for any or all of the following:

- Identification/authentication,
- Confidentiality,
- Message/transaction, and/or
- Non-repudiation.

**How does PKI satisfy those Business Environment needs?**
Large enterprises such as government, banks, and large commercial firms will develop trust models to easily incorporate PKI into everyday business use. Thus far, a significant number of PKI projects have been curtailed, revised, or temporarily shelved for reevaluation. The reasons most often given include the following:

- Immature technology,
- Insufficient planning and preparation,
- Underestimated scope,
- Infrastructure and procedural costs,
- Operational and technical incompatibilities, and/or
- Unclear cost-benefits.

**How well does PKI satisfy today’s open systems security needs?**
PKI is an evolving process. It has the fundamental strength, granularity, and flexibility required to support the security requirements, and is the best available alternative. However, it should be examined selectively by business process or application to determine whether there is sufficient ‘value-added’ to justify the direct and indirect cost associated with deployment.

**What about the cost?**
The most common approach to launching PKI is a pilot environment. Map the due diligence and procedural requirements against the culture of the organization. Look at the volatility of the certificates that will be issued. What is their life expectancy and need for modification? What is the growth curve for certificate use? How flexible is the design of the solutions being considered?
PKI price comparisons are not easy to forecast. Most vendors charge on a per-certificate, per-server, or per-application. They may add consulting fees and other service fees. However, research believes the models will change in favor of per-user or site licenses as vendors recognize customers' needs.

12.17 Middleware Questions
The following is a suggested list of questions that should be presented to vendors of middleware products:

Does the Vendor have a middleware product or have a partnership with an organization that does?
What types of middleware products are available (i.e. Message Oriented, RPC, ORB)?

Features
Are supporting GUI tools packaged with the product (i.e. administering, transaction/performance monitoring/reporting, debugging, testing), and how extensive are their capabilities?
How does the product guaranteed message delivery?
What communication standards does the product use?
message system - MTS, JMS, CICS, TP Monitor, VB
message format - request/response, conversational

Is there a message prioritization associated with the product, and how does it work? At what point can a message be recalled or cancelled, if at all? What supporting documentation comes with the product and in what format? Do add-on components exist that can enhance functionality, what do these components do, and how available are they?

Performance
Are there any known performance inhibitors such as different types configurations, number of interfaces, number of concurrent users, or number of transactions?
What does the product provide for load balancing (fault tolerance)?
How does the product maintain reliability (availability 24 x 7 x 365)?

How does the product handle errors or exceptions?
How scalable is the product?
Is the product capable of evaluating blocked network routes and rerouting messages in those instances?

Infrastructure Compatibility/Requirements
What types of network protocols can the product operate under?
Can the product operate using varying speeds (T1, frame relay, 56k)?
Can the product operate under Novell NetWare 5 and can it leverage the NetWare features such as Directory Services?
Does the product support Single-System Logins?

Ease of Use/Integration/Implementation
What software languages (C, C++, Java, ActiveX) is the product compatible with?
How flexible is the architecture of the product?
Can parameters such as queue size, buffer size, and number/length of time-outs modifiable?

Does the middleware supply standard message formats and if so, are they modifiable? Can the user create unique message formats, and how difficult would that be? How hard would it be to integrate this product with products from other vendors, and how would that affect performance, security, and product management?

**Security**
What methods does the product use to maintain secure message delivery?
What type of encryption does the product use?
Does the product incorporate access control lists or user permissions?

**Software Requirements**
What other software is required for using or supporting the product (i.e. Word Perfect, Adobe Acrobat Reader, web browser)?

**Hardware Requirements**
What types of hardware specifications are needed to run the product from both the client and server sides (i.e. RAM, disk space, CPU)?

**Cost of Ownership**
How much does the software cost?
How much does the licensing cost (single and multiple)?
How are upgrades and updates handled, and how much do they cost?
Is support included? If so, what does it entail (phone, website)?
If not, how much would support cost?
Is on-site help available for pre-implementation testing?
## 12.18 Server Hardware Evaluation Matrix

This matrix should be completed prior to the decision to purchase Server Hardware.

<table>
<thead>
<tr>
<th>Criteria Evaluation</th>
<th>Linux</th>
<th>??????</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support for Required Functionality</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Online Performance and Response Time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>High Availability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connectivity to TCP/IP Based Network</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalability for End-User and Application Growth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor Viability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor Hardware and Operating System Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Architectural Longevity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Systems and Network Management Software</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Systems Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Availability of Trained Administrators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISV Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware Vendor Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web-Based Technologies Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Save Smart Vendor</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Typical Video Logical Layout

Provider
QWest ISDN

Provider

QWest ATM
ATM or FR T-1

300 San Mateo
Switch
QWest ATM

Provider

OC 3

Provider

Regional JRF D Router

Detail Facility Setup

Alamogordo
Albuquerque
Albuquerque
Albuquerque
Albuquerque
Albuquerque
Artesia
Bernalillo
Carlsbad
Clovis
Deming
Española
Estancia
Farmington
Farmington
Ft. Blanton
Gallup
Grants
Hobbs
Las Cruces
Las Cruces
Las Cruces
Las Vegas
Las Lunas
Portales
Raton
Roswell
Santa Fe
Santa Fe
Silver City
Socorro
Springer
Taos
T or C
Tucumcari

Last Updated 031207
12.20 Technical Requirements for FACTS

CYFD has determined that child welfare data must be collected in an integrated fashion. Specifically, the information must be inclusive from the first contact with the child and his or her family through the time that the child is no longer involved with the child welfare system. This includes the provision of investigative services, pre-placement, preservation, preventive services, placement services, and completion of reunification services, adoption or the provision of independent living services to those children who cannot be reunified or adopted. Certain activities must be recorded over the period of involvement in order to assess the impact of intervention and program goals. Federal legislation, PL 96-272, requires states to implement an information system to track children in substitute care and to establish other child protections to qualify for enhanced Title IV-B funding. These and other federal dollars support the child welfare services being provided today to CYFD children.

The key for the Type Column is:

<table>
<thead>
<tr>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Consistency</td>
</tr>
<tr>
<td>D</td>
<td>Data</td>
</tr>
<tr>
<td>E</td>
<td>External</td>
</tr>
<tr>
<td>G</td>
<td>General System Design</td>
</tr>
<tr>
<td>O</td>
<td>Operational</td>
</tr>
<tr>
<td>P</td>
<td>Performance</td>
</tr>
<tr>
<td>S</td>
<td>Security</td>
</tr>
<tr>
<td>U</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>V</td>
<td>Development</td>
</tr>
<tr>
<td>W</td>
<td>Workflow</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>5</td>
</tr>
<tr>
<td>UD</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>7</td>
</tr>
<tr>
<td>Column</td>
<td>Row</td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>9</td>
</tr>
<tr>
<td>W</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>11</td>
</tr>
<tr>
<td>C</td>
<td>12</td>
</tr>
<tr>
<td>DC</td>
<td>13</td>
</tr>
<tr>
<td>D</td>
<td>14</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>---</td>
</tr>
<tr>
<td>W</td>
<td>15</td>
</tr>
<tr>
<td>U</td>
<td>16</td>
</tr>
<tr>
<td>UC</td>
<td>17</td>
</tr>
<tr>
<td>UC</td>
<td>18</td>
</tr>
<tr>
<td>DC</td>
<td>19</td>
</tr>
<tr>
<td>C</td>
<td>20</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
<tr>
<td>C</td>
<td>21</td>
</tr>
<tr>
<td>D</td>
<td>22</td>
</tr>
<tr>
<td>D</td>
<td>23</td>
</tr>
<tr>
<td>D</td>
<td>24</td>
</tr>
<tr>
<td>U</td>
<td>25</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
<tr>
<td>U</td>
<td>26</td>
</tr>
<tr>
<td>UC</td>
<td>27</td>
</tr>
<tr>
<td>U</td>
<td>28</td>
</tr>
<tr>
<td>U</td>
<td>29</td>
</tr>
<tr>
<td>D</td>
<td>30</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
<tr>
<td>D</td>
<td>31</td>
</tr>
<tr>
<td>E</td>
<td>32</td>
</tr>
<tr>
<td>D</td>
<td>33</td>
</tr>
<tr>
<td>D</td>
<td>34</td>
</tr>
<tr>
<td>W</td>
<td>35</td>
</tr>
<tr>
<td>U</td>
<td>36</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>---</td>
</tr>
<tr>
<td>O</td>
<td>37</td>
</tr>
<tr>
<td>W</td>
<td>38</td>
</tr>
<tr>
<td>UD</td>
<td>39</td>
</tr>
<tr>
<td>O</td>
<td>40</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>---</td>
</tr>
<tr>
<td>S</td>
<td>41</td>
</tr>
<tr>
<td>D</td>
<td>42</td>
</tr>
<tr>
<td>U</td>
<td>43</td>
</tr>
<tr>
<td>V</td>
<td>44</td>
</tr>
<tr>
<td>V</td>
<td>45</td>
</tr>
<tr>
<td>V</td>
<td>46</td>
</tr>
<tr>
<td>V</td>
<td>47</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
<tr>
<td>D</td>
<td>48</td>
</tr>
<tr>
<td>V</td>
<td>49</td>
</tr>
<tr>
<td>D</td>
<td>50</td>
</tr>
<tr>
<td>P</td>
<td>51</td>
</tr>
<tr>
<td>P</td>
<td>52</td>
</tr>
<tr>
<td>Type</td>
<td>#</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
<tr>
<td>P</td>
<td>53</td>
</tr>
<tr>
<td>G</td>
<td>54</td>
</tr>
<tr>
<td>C</td>
<td>55</td>
</tr>
<tr>
<td>S</td>
<td>56</td>
</tr>
<tr>
<td>E</td>
<td>57</td>
</tr>
</tbody>
</table>
Requirement: **AUDITING**

*Issue*: There is a need to provide information on who updated a specific row/record on any given date and time and indicate what changes were made.

*Discussion*: The Deputy Technical Manager raised the above issue at one of the Work Group sessions. In the past, there have been directives either by the Courts or Management to identify what changes a specific individual has made to a FACTS record, on what dates and at what times. Given this need, there is also a requirement to ‘back out’ a transaction that is deemed inaccurate, inappropriate.

*COMMENTS:*

In order to accomplish the above, it will be necessary to record transaction history.

The impact on disk storage and performance will be evaluated when we conduct capacity analysis.

*NOTE:*

Normal logging will be enabled/disabled at the RDBMS level.
12.21 ITS Firewall Standards

ITS requires an electronic firewall between its private network and any public or otherwise non-trusted network. This firewall must by default allow no electronic entry into itself or the ITS network from its public interfaces. All allowed entry must be manually configured. All entry originating outside of the firewall must be logged. Sessions originated on the private side of this firewall will be allowed out in accordance with Novell Border Manager or equivalent monitoring software.

Additional requirements for this firewall are:
1) Support Network Address Translation (NAT) to shield ITS internal network IP address from view of public networks.
2) Supports access based on TCP/IP addresses and/or port.
3) Supports JAVA blocking.
4) Supports URL filtering.
5) Supports SNMP management on the private network interface.
6) Supports 16 MB Token Ring, 10/100 Ethernet and Gigabit Ethernet.
7) Supports remote logging via TCP or UDP.
8) Supports E-mail and pager alarms based on single events or after a threshold is reached.
9) Provides built-in reports to display FTP and URL activity per user on a daily basis.
10) Supports IPSEC encryption.
11) Supports VPNs.
12) Supports a DeMilitarized Zone (DMZ). Access via a DMZ is by default refused; access must be manually configured.
12.22 Supporting Documentation for Technical Architecture Components

This Appendix provides in one place, references to all of the supporting documents that define the current CYFD Technical Architecture.

**Business Architecture**

G:\ADS\RELEASES\release v340\Topic Papers\Completed  
G:\ADS\RELEASES\release v3500\Topic Papers\Completed  
G:\ADS\RELEASES\release v3500\Requirements  
G:\ADS\RELEASES\release v3500\System Design\Change Request  
G:\ADS\RELEASES\release v3500\System Design\Change Request\FSD\CR800\R3500CR800-Child Care Eligibility.doc  
U:\FACTSBAT\ProdDocs

**FACTS Hardware and Network Architecture**

G:\TSS\Novell\Documentation\Netware 6 Information\NW6 Planning XLS\Novell Environment.xls  
G:\TSS\TeleCommDB\TeleComTracking2.mdb  
(Can Access Database)

**Data Architecture**

The Help Desk can place a copy of \Data Dictionary\dssNMDD.mdb - an Access Database that contains the data dictionary for FACTS, on your C: drive.

**Current Technical Environment Overview**

G:\Mgmt\Reporting\Inventory\Software Inventory.xls
12.23 Information Technology Change Control Committee Charter

1. Purpose
This charter defines the mission, objectives, organization and procedures of the Information Technology Change Control Committee (hereafter referred to as ITCCC). The ITCCC will report to the CYFD Information Technology Steering Committee.

2. Mission
To provide a forum for CYFD divisions to review, prioritize and coordinate software modifications to the Family Automated Client Tracking System (hereafter referred to as FACTS). This process is critical in allowing the Information Technology Services (hereafter referred to as ITS) to provide software modifications that are accurate, timely and meet all federal and state requirements.

3. Objectives
Software modifications to the FACTS online application are implemented based on the current release strategy. Batch reporting software can be implemented outside of a scheduled release. The planning process will be the same for both but the implementation plans may vary.

Providing CYFD agency direction to ITS regarding FACTS software modifications is critical. Modifications to FACTS must be prioritized to meet CYFD needs and meet any federal and/or state requirements.

3.1. Participate in the review of all current FACTS modifications/enhancements including FACTS Help Desk Incident Reports.

3.2. To review all pending FACTS Change Requests and determine priority.

3.3. Identify future FACTS Change Requests and determine priority. All new FACTS Change Requests will be initiated by the divisions through this committee. FACTS Change Requests will only be submitted to ITS through this committee.

3.4. Direct ITS to provide time estimates for FACTS Change Requests based on the change request priority.

3.5. Based on time estimates, determine the implementation plan for FACTS Change Requests. This process is critical in planning future enhancements/modifications to FACTS.
3.6. Advise IT Steering Committee on priorities defined for FACTS modifications/enhancements and FACTS Change Requests.

4. Organization and Procedures

4.1. Membership

4.1.1. Membership in ITCCC shall include representation from each division of CYFD.

4.1.2. Each division will designate one voting Representative, and one Alternate (who will act on behalf of the voting Representative when the voting Representative cannot attend).

4.1.3. Each Division or Division Representative will have one vote with the exception of the ITCCC Chair. ITCCC Chair shall be a non-voting member.

4.1.4. The ITCCC member shall act as the official division liaison with division authority to make decisions on voting issues.

4.1.5. Voting memberships may be extended to other agencies or entities as IT Steering Committee deems necessary or appropriate to fulfill its mission.

4.1.6. If voting membership results in an even number and a vote results in a tie, the ITCCC Chair shall cast the deciding vote.

4.2. Chair

4.2.1. The committee will be chaired by ITS.

4.2.2. Duties and responsibilities of the Chair (or the Chair’s designee) include;  
   4.2.2.1. Appoint a recording secretary from ITS, who shall be responsible for drafting official minutes of ITCCC meetings.
   4.2.2.2. Prepare an agenda, arrange a time, place and facilities for, and preside over ITCCC meetings. Deliver the meeting agenda to ITCCC members one week in advance of the scheduled meeting date.
   4.2.2.3. Update and maintain the FACTS Change Control list and deliver the updated list along with the agenda one week prior to the scheduled meeting date.
   4.2.2.4. Report to the IT Steering Committee on ITCCC consensus or result of voting actions.

4.3. Subcommittees

4.3.1. Subcommittees may be established as working groups at the direction of ITCCC.

4.3.2. Subcommittee chairs shall be elected by a simple majority of the voting membership present at a scheduled meeting or may be appointed by the ITCCC Chair.

4.3.3. Subcommittee chairs shall report to the ITCCC membership.
4.4. Charter Revisions
  4.4.1. Charter revisions shall require a two-thirds majority of those votes cast for approval.
  4.4.2. All Charter revisions shall require a one-month comment period.
  4.4.3. The proposed revised Charter shall be sent to the IT Steering Committee for final approval.

4.5. Procedural Issues
  4.5.1. ITCCC meetings will be held monthly.
  4.5.2. The ITCCC Chair may call a special session of the ITCCC Committee to address any critical issues identified by the ITCCC Committee members.
  4.5.3. ITCCC members shall excuse themselves from participation in issues that could be defined as a conflict of interest.
  4.5.4. ITCCC meetings are open to the public.

5. ITCCC Functions
The ITCCC functions shall include, but not be limited to advisory, coordination and administrative functions.

5.1. Advisory Functions
  5.1.1. Advise divisions on FACTS enhancements/modifications.
  5.1.2. Inform IT Steering Committee on ITCCC issues.
  5.1.3. Direct ITS on priority and direction of pending FACTS enhancements/modifications.
  5.1.4. Submit all FACTS Change Requests to ITS.
  5.1.4. Identify all resources necessary to implement FACTS enhancements/modifications. This includes ITS and non-ITS resources. Advise IT Steering Committee on resource issues.

5.2. Coordination Functions
  5.2.1. Appoint subcommittees to address ITCCC issues as deemed necessary by the ITCCC.

5.3 Administrative Functions
  5.3.1. Conduct regular meetings.
  5.3.2. Represent ITCCC activity to groups internal and external to CYFD as deemed necessary by the ITCCC.
12.24 Project Plan – Phase 2 – System Development Phase Plan

APPLICATIONS DEVELOPMENT SECTION
PROJECT PLAN

PHASE 2 - SYSTEM DEVELOPMENT PHASE PLAN
CR # XXX - PROJECT NAME

2.1 Objective
The objective of Phase 2 is to define the system development life cycle of the plan.

2.2 Discussion
The System Development Phase will consist of six (6) phases; each phase produces a deliverable that will provide input to other phases of the project. The phases are:

- Definition
- System Design
- Software Development
- System/Regression Test
- Acceptance Test
- Implementation

Primary objectives and secondary objectives are listed for each phase in the Detail Section.

2.3 Detail

2.3.1 Definition Phase
The users provide their requirements definition in this phase.

2.3.1.1 Primary Objectives

a) Requirements Definition
   The users will define the purpose of the project and the business rules required to implement the project. The business rules will be translated into the Requirements Definition document.
   **Deliverable:** Requirements Definition Matrix document signed by the authorized user(s).

b) Detailed Project Plan
   The Project Manager for ITS will create the Project Plan during this phase. The Project Plan will be updated throughout the project.
   **Deliverable:** The Project Plan Document.

c) Set Up Project Library
   ITS will create the necessary environment to store the documentation that will be used during the project.
   **Deliverable:** Project Library
12.25 System Design Documents
Online System Design Document Template

Double click the icon that follows to access the System Design Document template:

*System Design Form.doc*

Double click the icon that follows to access instructions for using the System Design Document template:

*System Design Form Instructions.doc*

Batch System Design Document Example
Double click the icon that follows to access the Batch System Design Document Example:

Go to pages 413–416 of the Operations Guide to get the above referenced document.
12.25 Technology Architecture – Phase I FACTS

CVS is a version control system, an important component of source configuration management (SCM).

Apache Ant – a Java-based build and deployment tool.

DEVELOPMENT

TEST

UAT

WEB SERVER

APP. SERVER

WEB SERVER

APP. SERVER

WEB SERVER

APP. SERVER

WEB SERVER

APP. SERVER

WEB SERVER

APP. SERVER

WEB SERVICES STACK

Apache Axis2 – an implementation of the SOAP ("Simple Object Access Protocol").

Web services and the J2EE support the distributed services.

JUDDI (pronounced "juh-di") is a universal description, discovery, and integration (UDDI) specification for web services.

JAXME 2 is an open source way to bind an XML schema to a representation in Java code.


The Apache JAXM PROJECT is a Java implementation of web services standard - WSDL/Soap/HTTP/URIs.

Apache WSIF is a robust Java implementation of the Web Services Resource Framework (WSRF) family of specifications.

Staging Server

Low Server

DEVELOPMENT FILESYSTEM

TEST FILESYSTEM

UAT FILESYSTEM

Database Server

W:\ITS Web Project\Documentation\Environment Documents\Visios
13 Revision History

As updates are made to the System Architecture Specification Document, a record is inserted into the table below to track changes. This table provides a revision history for the document. The “Description of Updates” column should contain the page number of the change and the reason for the change. The name of the person who made the change should be entered into the “Updated By” column.

<table>
<thead>
<tr>
<th>Approval Date</th>
<th>Updated By</th>
<th>Description of Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/9/2007</td>
<td>C. Spooner</td>
<td>The entire document was updated to make current.</td>
</tr>
<tr>
<td>October 19, 2007</td>
<td>C. Spooner</td>
<td>The entire document was updated to make current.</td>
</tr>
</tbody>
</table>
NOTE TO MYSELF:
PLACE IN APPROPRIATE PLACE .... January 18, 2006