

**Department of Administration, Information Resources Management Division
Enterprise Application Services -- Application Development Documentation Standards for
Statewide Systems**



Prepared by: Linda M. Campbell / Joe Adelman
Department of Administrative Services
Information Resources Management Division
Enterprise Application Services Section
Revision Date: August 23, 2005 (Section name change only)

Application Standards Table of Contents

1. EXECUTIVE SUMMARY	5
2. DESIGN PRINCIPALS FOR STATEWIDE SYSTEMS.....	5
3. SUPPORTED PLATFORMS FOR STATEWIDE SYSTEMS	5
3.1. DATABASES.....	5
3.1.1. <i>Mainframe</i>	5
3.1.2. <i>Mid-Range</i>	5
3.2. OPERATING SYSTEM.....	6
3.2.1. <i>Microsoft Windows</i>	6
3.2.2. <i>Unix</i>	6
3.2.3. <i>VMS</i>	6
3.3. INTERNET BROWSERS.....	6
3.3.1. <i>Internet Explorer</i>	6
3.3.2. <i>Netscape</i>	6
3.3.3. <i>Mozilla</i>	6
4. DEVELOPMENT METHODOLOGIES FOR STATEWIDE SYSTEMS.....	6
5. GENERAL DESIGN REQUIREMENTS :	7
5.1. DATABASE STRUCTURES	7
5.2. LEGACY FILE STRUCTURES	7
6. TECHNICAL DOCUMENTATION	7
6.1. PROJECT MANAGEMENT PLAN	7
6.1.1. <i>Description of the next phases after Requirements</i>	7
6.1.2. <i>Project Design Principles</i>	7
6.2. REQUIREMENTS DOCUMENTATION.....	7
6.2.1. <i>Executive Summary</i>	7
6.2.2. <i>Current Business and Systems Environment</i>	7
6.2.3. <i>Operational Implications</i>	7
6.2.4. <i>Project Scope</i>	8
6.2.5. <i>Describe existing functions the will be maintained and improved</i>	8
6.2.6. <i>Describe new functions that will be added</i>	8
6.2.7. <i>Regulatory/Statutory (Must have)</i>	8
6.2.8. <i>Business Need (Need to Have)</i>	8
6.2.9. <i>Electives (Would like to have)</i>	8
6.2.10. <i>Describe general requirements of application such as security, history, response time, etc</i>	8
6.2.11. <i>Functional Requirements Summary</i> :	8
6.2.12. <i>Data Requirements</i> :	8
6.2.12.2. <i>Data Migration and Transformation</i>	8
6.3. CONCEPTUAL DESIGN DOCUMENTATION	8
6.3.1. <i>Business Process Design</i>	8
6.3.2. <i>Conceptual Overview</i>	8
6.3.3. <i>Functional Components</i>	9
6.3.4. <i>Technical Supplements</i>	9
6.4. REQUIREMENTS TRACEABILITY DOCUMENTATION	9
6.4.1. <i>Requirements ↔ Design</i>	9
6.4.2. <i>Requirements ↔ Code</i>	9
6.4.3. <i>Design ↔ Code</i>	9

Enterprise Application Services - Application Development Documentation Standards

- 6.4.4. Requirements ↔ Test..... 9
- 6.5. FUNCTIONAL SPECIFICATIONS..... 9
 - 6.5.1. Purpose..... 9
 - 6.5.2. Workflow..... 9
 - 6.5.3. Function and Behavior 9
- 6.6. DETAIL DESIGN DOCUMENTATION..... 9
 - 6.6.1. Conceptual Low Level Architecture: 10
 - 6.6.2. Conceptual Data Model: 10
 - 6.6.3. Schema Design Diagram (Physical Data Model):..... 10
 - 6.6.4. Data Dictionary:..... 10
 - 6.6.5. Reports..... 10
 - 6.6.6. Screens..... 10
 - 6.6.7. Indexes 10
 - 6.6.8. Frequency 10
 - 6.6.11. Table..... 10
 - 6.6.12. Stored Procedures 10
 - 6.6.13. Triggers 11
 - 6.6.14. Functions 11
 - 6.6.15. Sequences..... 11
 - 6.6.16. Constraints..... 11
 - 6.6.17. Relationship Reports..... 11
 - 6.6.18. Table Indexes/Usage..... 11
 - 6.6.19. Data Integrity..... 11
- 6.7. TEST STRATEGY DOCUMENTATION..... 12
 - 6.7.1. Unit Testing 12
 - 6.7.1.1. Development Environment (Developer Schema's). The database objects in the Development Test Environment will be used for testing as much as possible..... 12
 - 6.7.2. Features to be Tested..... 12
 - 6.7.3. Application Functional Testing..... 12
 - 6.7.3.1. Test analysts will do Application functional testing in the Acceptance Test Environment. Formal testing plans and scripts will be created using written application documentation to insure that the application fulfils **all** of the requirements of application. 12
 - 6.7.4. Ad-hoc Testing..... 12
 - 6.7.4.1. Test analysts and end users will do Ad Hoc testing in the Acceptance Test Environment. Test plans will be created and used as a guide to the testing process for end users and test analysts. 12
 - 6.7.5. Interoperability Testing 12
 - 6.7.5.1. Test analysts and end users will do Interoperability testing in the Acceptance Test Environment. Formal test plans and scripts will be created and used by the test analysts..... 12
 - 6.7.6. Stress Testing..... 12
 - 6.7.6.1. Test analysts will do Stress testing in the Acceptance Test Environment. Formal test plans and scripts will be created and used by the test analysts. 12
 - 6.7.7. User Friendliness Testing..... 12
 - 6.7.7.1. Test analysts and end users will do User friendliness testing in the Acceptance Test Environment. Test plans will be created and used as a guide to the testing process for end users and test analysts..... 12
 - 6.7.7.2. Standards Testing 12
 - 6.7.7.2.1. Test analysts will do coding etc. testing all standards such as GUI look and feel, database, naming, in the Acceptance Test Environment. Formal testing plans will be created and used by the test analysts..... 12
 - 6.7.8. Regression Testing 12
 - 6.7.9. Schedule/Testing Dates..... 12
 - 6.7.10. Item Pass/Fail Criteria 12
 - 6.7.11. Problem Reporting and Resolution..... 12
 - 6.7.12. Lead Responsibilities 12
 - 6.7.13. Test Deliverables 13
 - 6.7.14. Environmental Needs..... 13
 - 6.7.15. Staffing..... 13
 - 6.7.16. Approvals..... 13
- 7. DATA CONVERSION..... 13

Enterprise Application Services - Application Development Documentation Standards

7.1.	READ ONLY REQUIREMENT.....	13
7.1.1.	<i>Direct real time read of legacy data at the source where neither business or technical issues are comprised. (preferred method).....</i>	13
7.1.2.	<i>Maintain on source of data.....</i>	13
7.1.3.	<i>Replication of the data.....</i>	13
7.1.4.	<i>All shared data, captured in new applications will be bridged back to Legacy File structures making it available to Legacy systems. One source of data MUST be maintained.....</i>	13
7.2.	UPDATE REQUIREMENT.....	13
7.2.1.	<i>Direct real time update access capability of legacy data at the source where neither business or technical issues are compromised. (preferred method).....</i>	13
7.2.2.	<i>Provide backward real time bridge.....</i>	13
7.2.3.	<i>Provide batch update capability.....</i>	13
7.2.4.	<i>Provide for manual reconciliation/maintenance activities.....</i>	13
7.2.5.	<i>Storing of unsynchronized data.....</i>	13
8.	INTEGRATION.....	13
8.1.	INTERFACES.....	13
8.2.	MIGRATION.....	13
9.	DEVELOPMENT TOOLS :.....	13
9.1.	MAINFRAME.....	14
9.1.1.	<i>Cobol.....</i>	14
9.1.2.	<i>CICS.....</i>	14
9.1.3.	<i>JCL.....</i>	14
9.1.4.	<i>Easytrieve.....</i>	14
9.2.	OPEN SYSTEMS.....	14
9.2.1.	<i>Java.....</i>	14
9.2.2.	<i>ColdFusion.....</i>	14
9.2.3.	<i>HTML.....</i>	14
9.2.4.	<i>Javascripting.....</i>	14
9.2.5.	<i>PSP.....</i>	14
9.2.6.	<i>SQL.....</i>	14
9.2.7.	<i>PL/SQL.....</i>	14
10.	DATABASE.....	14
10.1.1.	14
11.	PRODUCTION ENVIRONMENT MAINTENANCE.....	14
11.1.	HARDWARE.....	14
11.2.	SOFTWARE.....	14
11.2.1.	<i>System overview data flowchart and description.....</i>	14
11.2.2.	<i>Major Process/Function data flowchart and description.....</i>	14
11.2.3.	<i>Overview and diagram describing the application modules and functionality.....</i>	14
11.2.4.	<i>Directory/Subdirectory/file structure for executables, dlls, and data for developmental and non-developmental software.....</i>	14
11.2.5.	<i>Identify developmental vs. non-developmental software.....</i>	14
11.2.6.	<i>Application specific standards.....</i>	14
11.3.	NETWORK.....	14
11.3.1.	<i>Network expectations should be adequately described and depicted.....</i>	14
11.4.	DEVELOPMENT TOOLS.....	15
11.4.1.	<i>Tool Generated Code – General concerns.....</i>	15
11.4.2.	<i>Hand Generated Code Standards.....</i>	15
11.4.4.	<i>Code Generators/CASE tools/Editors.....</i>	15
11.4.5.	<i>Compilers/linkers/build tools.....</i>	15
11.5.	OPERATIONAL TOPICS.....	15
11.5.2.	<i>Run schedule.....</i>	15
11.5.6.	<i>Backup and recovery procedures.....</i>	15
11.5.7.	<i>Re-start instructions.....</i>	15

11.5.8. Maintenance schedule.....	15
12. CONFIGURATION MANAGEMENT	15
12.1. VERSION CONTROL PROCESS	15
12.2. SERVICE REQUEST PROCESS.....	15
12.3. IMPLEMENTATION PLAN.....	15
13. NAMING CONVENTIONS.....	15
13.1. OPERATING SYSTEM PROJECT FILES	15
13.2. SOURCE CODE	15
14. CONTINGENCY PLAN.....	16
14.1. RISKS.....	16
14.2. EFFECT TO CUSTOMER	16
14.3. CONTACT INFORMATION.....	16
14.4. BACKOUT AND RECOVERY TASKS.....	16
15. KNOWLEDGE TRANSFER.....	16

1. **Executive Summary**

This document outlines the general design principles and documentation standards required internally to IRMD and of vendors contracted to develop state-wide systems for Department of Administrative Services (DAS) IRMD. The role of Enterprise Application Services will be defined as oversight to ensure the standards and procedures are adhered to by vendors contracting with the state. Their participation is mandatory on any application development projects that falls within the definition of statewide systems defined below. According to DAS policy the role of Enterprise Application Services is as follows :

POLICY: Information Resources Management Division, Enterprise Application Services Section (IRMD EAS) is responsible for the development and management of all computer applications that are defined as statewide DAS systems.

“Statewide DAS systems” are defined as:

- affecting all, or a large portion of, state agencies and stakeholders;
- requiring direct or indirect input from, and output to, many agencies;
- requiring reconciliation or interfacing to one of the existing major DAS systems, such as the Oregon State Payroll System, Statewide Financial Management System, Automated Budget System or Position Information Control System ;
- requires interfacing from or to other state agency’s systems such as Personal and Position Database System;
- providing access to citizens; or resulting from actions of the Information Resources Management Council.

2. **Design Principals for Statewide Systems**

These design principals have been developed to guide vendors in the design of new applications and their related data interfaces to and from legacy statewide systems. This strategy minimizes inconvenience to the agencies and promotes coherence in the automation process.

- 2.1. Development of statewide applications that will require accessing (reading and/or updating) existing Legacy application data, will be measured against the following principles and require IRMD consensus:
- 2.2. Please note the integration/migration strategy of the application may necessitate a method listed below that is not the preferred option.

3. **Supported Platforms for Statewide Systems**

3.1. **Databases**

- 3.1.1. Mainframe
 - 3.1.1.1. DB2
 - 3.1.1.2. IMS
 - 3.1.1.3. Oracle 9iAs

- 3.1.2. **Mid-Range**
 - 3.1.2.1. DB2
 - 3.1.2.2. Oracle
 - 3.1.2.3. SqlServer

3.2. Operating System

- 3.2.1. Microsoft Windows
- 3.2.2. Unix
- 3.2.3. VMS

3.3. Internet Browsers

- 3.3.1. Internet Explorer
- 3.3.2. Netscape
- 3.3.3. Mozilla

4. Development Methodologies for Statewide Systems

The following chart identifies the appropriate methodology approach for different projects. Any development project that meets the definition of ‘statewide’ under Section I of this document must follow the approach outlined for Medium/Large Application Development under the ‘Project Type’ column.

Project Type	Project Definition	Expected Deliverables			
		Initiation & Definition	Planning	Execution & Control	Closure
Production Support	Production stopped Bad data or erroneous information with high impact to customer No work around Preventative maintenance or small single enhancement Small software or hardware enhancements / upgrades with minimal impact to work unit				Resolution recap SFMS, OSPS, PPDB change control ticket closed
Evaluation	Preliminary assessment of potential projects Technology assessment Problem assessment BPI	<u>Project Plan *</u> <u>Project Status Report</u>			Evaluation Results Document (STP, BPI, White Paper)
Technology	Large hardware installations Major software changes / upgrades Significant changes to infrastructure Corporate moves Training	<u>Budget Document</u>	<u>GAD Project Status Report</u> <u>Timeline</u>	<u>Test Plan</u> <u>Training Strategy</u>	Post Implementation Review Document
Small Application Development (RAD)	No controversial issues Confined to a single development platform Minimal impact to affected work units work flow Prototyping used Training	<u>Project Charter Document</u>		<u>Updated Functional /Technical Specs</u> <u>BP System Documentation</u>	
Medium / Large Application Development	May have multiple components BPI or Work Flow Analysis required Conversion component Interface component Impact to affected work units May involve multiple development platforms	<u>GAD Requirement Document</u> <u>Project Plan *</u> <u>Project Status Report</u>	<u>Conceptual Design</u> Detail Design QA	<u>Functional /Technical Specs</u> <u>Constructed System</u> <u>Test Strategy</u> <u>Training Strategy</u> <u>BP System Documentation</u> <u>Implementation Plan</u>	Post Implementation Review Document

5. General Design Requirements:

5.1. Database Structures

5.1.1. New Statewide applications will be housed and maintained in relational database structures. See supported platforms.

5.2. Legacy File Structures

5.2.1. New data not currently defined in Legacy file structures will reside only in the new file structures and will not be added to legacy file structures.

5.2.2. Agencies will not be forced to maintain the same data in new and old applications. As new applications take ownership for data on-line, Legacy applications will be retired or modified to « turn-off » all update for shared items.

6. Technical Documentation

6.1. Project Management Plan

6.1.1. Description of the next phases after Requirements.

6.1.2. Project Design Principles

6.2. Requirements Documentation

The following components must be included in the requirements document :

6.2.1. Executive Summary.

6.2.1.1. Provides the reader with a narrative with a summary of the business scope of the project and the purpose of the requirements phase

6.2.2. Current Business and Systems Environment.

6.2.2.1. This is an analysis of current automated Environment. Describes the shortcomings of the current environment – those things that necessitate the project. Describe in terms of system and data. Describe on-line transactions and their business functions, batch processing and corresponding business functions, business and production reporting including transaction counts, number of users, etc.

6.2.3. Operational Implications

6.2.3.1. Description of any operational implications such as new hardware or software that may will be required. Consistency of schedules for interfacing systems.

- 6.2.4. Project Scope.
- 6.2.5. Describe existing functions the will be maintained and improved.
- 6.2.6. Describe new functions that will be added.
- 6.2.7. Regulatory/Statutory (Must have)
- 6.2.8. Business Need (Need to Have)
- 6.2.9. Electives (Would like to have)
- 6.2.10. Describe general requirements of application such as security, history, response time, etc.
- 6.2.11. Functional Requirements Summary:
- 6.2.12. Data Requirements:
 - 6.2.12.1. Identify data to be included at a summary level such as security, calculation, employee, agency, etc.

6.2.12.2. Data Migration and Transformation

6.2.12.2.1. This section shall include the data migration map and data migration/transformation plan. This should include proposed options for bad data. This section includes the plans for taking existing data and transforming /migrating into the correct values/format of the new application. If information exists in electronic format that will need to be converted for use by the new application, information about data conversion/archiving will also be included. A cross-reference may be used to indicate how the existing data will be used to update the new data elements. If the application is replacing an existing one which will be disabled, any data which will not be retained or archived, and therefore not easily accessed, should be explicitly listed, carefully reviewed and signed off by customer representative and Enterprise Application Services.

6.2.12.3. Data Conversion :

6.2.12.3.1. Conversion strategy should include the ‘clean-up’ of data that is incorrect or incomplete. This needs to be done before it is moved to the new application. Incorrect, incomplete or missing data exists. The conversion effort will be to research that data, define the appropriate resolution for blank or erroneous data and implement that resolution.

6.2.12.3.2. Interface Strategy

6.2.12.3.2.1. Describe any system processes that will require interfaces with legacy systems. Also describe any legacy processes that may need to be replaced, optimized and/or enhanced.

6.3. Conceptual Design Documentation

- 6.3.1. Business Process Design
 - 6.3.1.1. Describe changes to the current work flow in the business areas that will result. Describe opportunities to streamline workflow or increase efficiency of staff time. Discuss any legacy functions that will be retired.
- 6.3.2. Conceptual Overview
 - 6.3.2.1. Provide a two-part overview: one part graphic, one part narrative.
 - 6.3.2.2. The graphic overview should portray the system. The graphic should include the functional components detailed in section 2.
 - 6.3.2.3. The narrative should provide a narrative that describes what the system will do. A high level description of the system.

6.3.3. Functional Components

6.3.3.1. Describe each functional component with purpose that describes the functionality to the extent that a new maintenance developer or analyst could determine if a functional deficiency is a bug or an enhancement.

6.3.3.2. Functional components should include windows/screens, reports and major batch processes such as conversions, interfaces, major changes to legacy software, etc.

6.3.4. Technical Supplements

6.3.4.1. Logical Data Model

6.3.4.2. Development strategies/platforms

6.3.4.3. Additional security functions not described above

6.3.4.4. Commentions to other applications

6.3.4.5. Legacy optimization approach if applicable

6.3.4.6. Hardware considerations

6.3.4.7. Volume estimates (e.g. DASD, printing, LAN traffic)

6.4. Requirements Traceability Documentation

6.4.1. Requirements \leftrightarrow Design

6.4.1.1. Design Phase: Provide a clear traceability to the specific requirement or requirements from a given segment of the design document.

6.4.2. Requirements \leftrightarrow Code

6.4.2.1. Construction Phase: Provide a clear traceability from the code to the requirements.

6.4.3. Design \leftrightarrow Code

6.4.3.1. Construction Phase: Provide a clear traceability from the code to the design.

6.4.4. Requirements \leftrightarrow Test

6.4.4.1. Acceptance Test Phase: Provide a clear traceability from all test documents to the requirements

6.5. Functional Specifications

6.5.1. Purpose

6.5.1.1.1. Provide a high level narrative that describes the purpose of the utility in a manner that someone new to the system could pick it up and review with the source code.

6.5.2. Workflow

6.5.2.1.1. Describe in narrative format the workflow process. Describe input process and output processes. Provide graphic of screens and describe what data is included and how they are activated. Data Flow Diagrams here?

6.5.3. Function and Behavior

6.5.3.1.1. Describe primary functions of each screen and unique behaviors.

6.6. Detail Design Documentation

Technical Design Specifications should convey the inter workings of the system from end to end. Included in this are:

Enterprise Application Services - Application Development Documentation Standards

- 6.6.1. Conceptual Low Level Architecture:
 - 6.6.1.1.1. For process modeling, this section decomposes and describes subprocesses (functions). For object modeling, this section identifies and describes object (class) methods and messages. Also include:
- 6.6.2. Conceptual Data Model:
 - 6.6.2.1.1. This section includes a graphical representation and a textual description of the logical database structures. The relationships between tables and entities for the overall application should be clear. The table depictions should be populated with data from the data dictionary. This representation should include relationships between tables and external entities as well. Each textual description should include a definition of the entity/table and their relationships.
- 6.6.3. Schema Design Diagram (Physical Data Model):
 - 6.6.3.1.1. This section shall include a graphical representation of the physical structure of the database tables and a textual description of those tables.
- 6.6.4. Data Dictionary:
 - 6.6.4.1. This section shall include a list of all data elements included in the physical schema. At a minimum, each data element should contain the following. A more complete listing is found under section F.
 - 6.6.4.2. Data Type
 - 6.6.4.3. Data Format/Length
- 6.6.5. Reports
 - 6.6.5.1. Crosswalk to program names and vice versa.
 - 6.6.5.2. Crosswalk to table/element origin/calculation and vice versa.
 - 6.6.5.3. List and sample of all reports.
- 6.6.6. Screens
 - 6.6.6.1. Crosswalk to Table/Elements and vice versa.
- 6.6.7. Indexes
 - 6.6.7.1. Index of jobs/scripts.
 - 6.6.7.2. Index of files and where used.
 - 6.6.7.3. Index of Scripts and how used.
- 6.6.8. Frequency
 - 6.6.8.1.
- 6.6.9. Synonyms**
 - 6.6.9.1. Private Synonyms
 - 6.6.9.2. Public Synonyms
- 6.6.10. Specifications/Algorithms for Derived Data
- 6.6.11. Table
 - 6.6.11.1. Description
 - 6.6.11.2. Initial size/potential size
 - 6.6.11.3. Attributes
 - 6.6.11.4. Description
 - 6.6.11.5. Size
 - 6.6.11.6. Valid values
 - 6.6.11.7. Usage
 - 6.6.11.8. Source
- 6.6.12. Stored Procedures
 - 6.6.12.1. Description
 - 6.6.12.2. In/out parameters
 - 6.6.12.3. Packages should be designed to contain procedures that perform the same category of tasks. For example all the integrity procedures should be put in an integrity package.
 - 6.6.12.4. Do not write a procedure to enforce the referential integrity of a foreign key when a constraint can serve the same purpose
 - 6.6.12.5. Avoid sending anonymous PL/SQL blocks from application code to the database

Enterprise Application Services - Application Development Documentation Standards

6.6.12.6. By moving PL/SQL out of the application into database stored procedures, unnecessary compilation of the PL/SQL code is avoided because the code is only compiled once when the procedure is initially created. When PL/SQL code is imbedded in application code and sent to the database in the form of an anonymous PL/SQL block, it must be compiled every time it is sent.

6.6.12.7. PL/SQL code is more likely to be duplicated when it is imbedded in application code and cannot be called from other code modules.

6.6.13. Triggers

6.6.13.1. Description

6.6.13.2. Triggers and procedures must be designed to perform a single task. Do not define long procedures or triggers with several distinct sub-tasks. This is to prevent the sub-task functionality from becoming duplicated in other triggers and procedures.

6.6.13.3. Do not define triggers or procedures that duplicate functionality already provided by a feature of ORACLE. Functionality provided by ORACLE is always more efficient than functionality provided by an application developer.

6.6.13.4. Do not define triggers or procedures that duplicate the functionality already provided by another trigger or procedure.

6.6.14. Functions

6.6.14.1. Description

6.6.14.2. In/out parameters

6.6.14.3. A stored function should perform a single unit of work. A series of tasks that achieve one result defines a unit of work. The advantages of this approach are modular code, streamlined processing performance for applications, simpler code maintenance and implementation of future enhancements.

6.6.14.4. All stored functions will return a consistent number of arguments in a result set. That is, do not return five columns if it is a Tuesday, and six columns the rest of the week.

6.6.14.5. All stored functions will return a consistent data type in a result set. That is, do not return five numeric columns if it is a Tuesday and five character columns the rest of the week

6.6.15. Sequences

6.6.15.1. Description

6.6.15.2. Usages

6.6.16. Constraints

6.6.16.1. Description

6.6.16.2. Usages

6.6.17. Relationship Reports

6.6.17.1. Program to table and element crosswalk (CRUD diagram) relationships and vice versa reports?

6.6.17.2. Any tools to automatically generate this relationship report?

6.6.18. Table Indexes/Usage

6.6.18.1.

6.6.19. Data Integrity

6.6.19.1. This section discusses how data integrity will be maintained within the data base. The following should be addressed:

6.6.19.2. Validation Procedures

6.6.19.3. Referential Integrity Rules

6.6.19.4. Approaches to Enforcing Business Rules

6.7. Test Strategy Documentation

Summarize the approach and tasks necessary to successfully complete the system test for this release. Include the tasks associated with the preparation and execution of the system testing; resource requirements and environment needs; testing schedule and responsible parties.

6.7.1. Unit Testing

6.7.1.1. Development Environment (Developer Schema's). The database objects in the Development Test Environment will be used for testing as much as possible

6.7.2. Features to be Tested

6.7.2.1. State in narrative form the general features to be tested. List individual components to be tested, i.e. screens, batch programs, interfaces, output documents.

6.7.3. Application Functional Testing

6.7.3.1. Test analysts will do Application functional testing in the Acceptance Test Environment. Formal testing plans and scripts will be created using written application documentation to insure that the application fulfills all of the requirements of application.

6.7.4. Ad-hoc Testing

6.7.4.1. Test analysts and end users will do Ad Hoc testing in the Acceptance Test Environment. Test plans will be created and used as a guide to the testing process for end users and test analysts.

6.7.5. Interoperability Testing

6.7.5.1. Test analysts and end users will do Interoperability testing in the Acceptance Test Environment. Formal test plans and scripts will be created and used by the test analysts.

6.7.6. Stress Testing

6.7.6.1. Test analysts will do Stress testing in the Acceptance Test Environment. Formal test plans and scripts will be created and used by the test analysts.

6.7.7. User Friendliness Testing

6.7.7.1. Test analysts and end users will do User friendliness testing in the Acceptance Test Environment. Test plans will be created and used as a guide to the testing process for end users and test analysts.

6.7.7.2. Standards Testing

6.7.7.2.1. Test analysts will do coding etc. testing all standards such as GUI look and feel, database, naming, in the Acceptance Test Environment. Formal testing plans will be created and used by the test analysts.

6.7.8. Regression Testing

6.7.8.1.1. State in narrative form the amount of regression testing that will be performed, and who is responsible for the execution of the regression test(s).

6.7.9. Schedule/Testing Dates

6.7.9.1.1. State begin and anticipated end dates of all test phases, include integration, stress and acceptance phases

6.7.10. Item Pass/Fail Criteria

6.7.10.1.1. Identify the pass/fail criteria for the items in the release.

6.7.11. Problem Reporting and Resolution

6.7.11.1.1. Identify in detail the agreed upon problem reporting and tracking method. Identify in detail the problem analysis and problem resolution method. Include responsibilities and authorities for each team (technical and test execution) for each area.

6.7.12. Lead Responsibilities

6.7.12.1.1. Identify the testing leads and list each lead's responsibilities during the test phase.

- 6.7.13. Test Deliverables
 - 6.7.13.1.1. Identify all the supporting deliverables for the test phase. For example, this document, test plans, problem resolution reporting, metric.
- 6.7.14. Environmental Needs
 - 6.7.14.1.1. Identify the environment needs to support the test phase, both physical and technical.
- 6.7.15. Staffing
 - 6.7.15.1.1. Identify the project staff required for successful test execution. Include both project team members and business partners.
- 6.7.16. Approvals
 - 6.7.16.1.1. Identify the method for reaching the freeze code decision.

7. Data Conversion

Data Conversion strategy should include the ‘clean-up’ of data that is incorrect or incomplete. This needs to be done before it is moved to the new application. Incorrect, incomplete or missing data exists. The conversion effort will be to research that data, define the appropriate resolution for blank or erroneous data and implement that resolution. Development of statewide applications that will require accessing (reading and/or updating) existing Legacy application data, will be measured against the following principles and require IRMD consensus:

7.1. Read Only Requirement

- 7.1.1. Direct real time read of legacy data at the source where neither business or technical issues are comprised. (preferred method)
- 7.1.2. Maintain on source of data.
- 7.1.3. Replication of the data
- 7.1.4. All shared data, captured in new applications will be bridged back to Legacy File structures making it available to Legacy systems. One source of data MUST be maintained.

7.2. Update Requirement

- 7.2.1. Direct real time update access capability of legacy data at the source where neither business or technical issues are compromised. (preferred method).
- 7.2.2. Provide backward real time bridge
- 7.2.3. Provide batch update capability
- 7.2.4. Provide for manual reconciliation/maintenance activities
- 7.2.5. Storing of unsynchronized data.

8. Integration

8.1. Interfaces

- 8.1.1. In accordance with design principle – only a single point of entry must be supported. All current functionality must be maintained and synchronization of old and new data files is mandatory.

8.2. Migration

9. Development Tools:

9.1. Mainframe

- 9.1.1. Cobol
- 9.1.2. CICS
- 9.1.3. JCL
- 9.1.4. Easytrieve

9.2. Open Systems

- 9.2.1. Java
- 9.2.2. ColdFusion
- 9.2.3. HTML
- 9.2.4. Javascripting
- 9.2.5. PSP
- 9.2.6. SQL
- 9.2.7. PL/SQL

10.Database

- 10.1.1.

11.Production Environment Maintenance

11.1. Hardware

- 11.1.1. Environment adequately described and depicted as it pertains to the Application Server, Web Server and Database Server

11.2. Software

- 11.2.1. System overview data flowchart and description.
- 11.2.2. Major Process/Function data flowchart and description.
- 11.2.3. Overview and diagram describing the application modules and functionality
- 11.2.4. Directory/Subdirectory/file structure for executables, dlls, and data for developmental and non-developmental software.
- 11.2.5. Identify developmental vs. non-developmental software.

11.2.6. Application specific standards

- 11.2.6.1. Program naming conventions
- 11.2.6.2. File naming conventions
- 11.2.6.3. Report naming conventions
- 11.2.6.4. Program Specific
- 11.2.6.5. Variable naming conventions
- 11.2.6.6. Internal module naming conventions

11.3. Network

- 11.3.1. Network expectations should be adequately described and depicted.

11.4. Development Tools

11.4.1. Tool Generated Code – General concerns

- 11.4.1.1. What ability to comment does the tool have?
- 11.4.1.2. Does commenting exist in generated code?
- 11.4.1.3. Could generated code be maintained if tool were unavailable? (Gain an understanding of the extent of dependency on tool.)
- 11.4.1.4. Accessibility/Organization of Generated Code via the tool?

11.4.2. Hand Generated Code Standards

- 11.4.2.1. Maintenance log box at the start of the program that would include the following: date, programmer, work request number, brief description of the change.
- 11.4.2.2. All scripts/programs to follow the same general documented processing (i.e. Paragraph and variable names).
- 11.4.2.3. Description of all inputs
- 11.4.2.4. Description of all outputs
- 11.4.2.5. File to function/subroutine organization
- 11.4.2.6. Header information for each file and function/subroutine
- 11.4.2.7. Inline comments sufficient enough to describe what code segment accomplishes
- 11.4.2.8. Inline comments sufficient enough to identify beginning and end of structures (if then else, Do loops, etc)
- 11.4.2.9. Indentation
- 11.4.2.10. Compliance with a coding methodology (OO or Structured)
- 11.4.2.11. All commented out code has been removed
- 11.4.2.12. Readability

11.4.3. Describe all tools used to develop/maintain/operate the system identified such as:

- 11.4.4. Code Generators/CASE tools/Editors
- 11.4.5. Compilers/linkers/build tools

11.5. Operational Topics

11.5.1. Describe any maintenance activities that need to be run.

- 11.5.2. Run schedule
- 11.5.3. Dailies?**
- 11.5.4. Weeklies?**
- 11.5.5. Monthly?**
- 11.5.6. Backup and recovery procedures
- 11.5.7. Re-start instructions
- 11.5.8. Maintenance schedule.

12. Configuration Management

12.1. Version Control Process

12.2. Service Request Process

12.3. Implementation Plan

- 12.3.1. The implementation plan needs to identify date, detail task list in sequence of processes necessary to implement and responsible person. Move to production should also include time estimates and approximate start times of any production jobs that need to be executed. Include appropriate contact information.

13. Naming Conventions

13.1. Operating System Project Files

13.2. Source Code

14. Contingency Plan

14.1. Risks

- 14.1.1. Identify all risks, business and technical as they relate to the contingency plan.

14.2. Effect to customer

- 14.2.1. Address business affects to customer

14.3. Contact information.

- 14.3.1. Include appropriate personnel to be notified.

14.4. Backout and recovery tasks

- 14.4.1. List all jobs and activities in appropriate sequence necessary to backout implementation changes and recover environment

15. Knowledge Transfer

15.1. Purpose

- 15.1.1. The purpose of Knowledge Transfer activities is to reduce future dependence on the vendor to make modifications and enhancements to Statewide systems. This assists the state in gaining expertise and taking ownership of it's systems.
- 15.1.2. Expertise in the application is not an immediate achievable goal of the knowledge transfer process. This is only acquired by those who either build the application or maintain the application over a period of time. It does provide for hands on experience for those individuals who will be responsible for maintaining the application. This way they become familiar with the interworkings of the technical processes from the vendor who built the application. Also, it is intended to help 'bridge the gap' between the documentation and the application. The best documentation cannot compensate for staff expertise on the application.

15.2. Timing

- 15.2.1. This should occur as each of the major portions of the application is turned over for acceptance testing. This provides an opportunity for the State IT staff to review the code and become familiar with portions of the application incrementally.

15.3. How do you know when KT has been achieved?

- 15.3.1. This is difficult to measure at best. There are several skill sets that will need to be acquired in order to successfully support a statewide application:
- 15.3.2. Solid technical skills in the development software in order to confidently make program changes.
- 15.3.3. Solid understanding of the architecture of the application, table structures, scripts, etc.
- 15.3.4. Solid understand of the application at a functional level, what processes are performed, how and when.
- 15.3.5. To order to achieve a level of expertise in these skills, it is critical that solid technical documentation be available to assist IT staff during this process.

15.4. KT plan:

- 15.4.1. The vendor must submit a detailed KT plan for a complete turnover of the application. Included in this plan are:
- 15.4.2. The designated state IT person must be considered as part of the project team and should be considered as a resource to the vendor in the project plan.
- 15.4.3. If the designated state IT person must does have the current technical skill set required to maintain the application, then the vendor must provide a schedule and approach for recommended training. The vendor must provide for additional training through hands on application work.
- 15.4.4. Training the State IT person in the 'skills' required is not part of knowledge transfer.

- 15.4.5. The designated state IT person should participate in Requirements and Design sessions and have input into the Design philosophy
- 15.4.6. An Overall system overview should be provided to the state IT resources
- 15.4.7. The designated state IT person should be involved in some aspect of the coding process at the direction of the contractor. This provides a 'hands on' effort that allows the State IT staff to gain further exposure to the application. This effort could be achieved either through new development or bug/maintenance work.
- 15.4.8. For each major process/function within the application, the vendor must provide to the State IT resource the following reviews:
- 15.4.9. Related screens and functionality as it relates to the technical processes 'behind' the scenes.
 - 15.4.9.1.1. Database
 - 15.4.9.1.2. ERD review
 - 15.4.9.1.3. related documentation
 - 15.4.9.1.4. Coding and testing process utilized for problem resolution
 - 15.4.9.1.5. Special processing

16. The vendor must provide complete technical documentation as outlined in Section IV of this document.

17. NOTES:

- 17.1. All documentation reviews should involve the the application development staff, Unix system administrator and physical database administrator as required.
- 17.2. The review of each specific application area should include sample code walk throughs and discussions.
- 17.3. The KT process should be broken into logical parts, allowing time for the State IT staff to review the pertinent materials and then have the ability to return with questions and further clarification of a process.