State of Michigan

(Insert System or Project Name Here)

Functional Design Document

# General Information

|  |  |  |  |
| --- | --- | --- | --- |
| System or Project ID/Acronym: |  | Creation Date: |  |
| Client Agency: |  | Modification Date: |  |
| Author(s): |  | DTMB Authorized by: |  |

# Privacy Information

This document may contain information of a sensitive nature. This information should not be given to persons other than those who are involved with this system/project or who will become involved during its lifecycle.

# Change Control

| Revision Date | Author | Section(s) | Summary |
| --- | --- | --- | --- |
| 1/3/2019 | Nancy Ziemski | Added 2. 1 Context Diagram  | (Added) Conceptual Level DiagramAdded narrative to describe what is needed. EIM Enhancement. |
| 1/3/2019 | Nancy Ziemski | Added 2.2 System Context Diagram | Added narrative to describe what is needed. EIM Enhancement. |
| 1/3/2019 | Nancy Ziemski | Added Section 2.2.1 System Level (1) and 2.2.2 System Level (2) | (Updated) Data Flow Diagrams.Added narrative to describe the needed data flow diagrams. EIM Enhancement. |
| 1/3/2019 | Nancy Ziemski | Updated section 3.1. Data Specification | Added narrative to describe what is needed for this section. EIM Enhancement. |
| 1/3/2019 | Nancy Ziemski | Updated section 3.1.1 Logical Data model | Added narrative to describe what is needed for this section. EIM Enhancement. |
| 1/3/2019 | Nancy Ziemski | Added 3.1.3 Data Share Requirements | Added narrative to describe what is needed for this section. EIM Enhancement. |
| 1/3/2019 | Nancy Ziemski | Updated 3.2 Human Interface Design | Added narrative to describe what is needed for this section. EIM Enhancement |
| 1/3/2019 | Nancy Ziemski | Updated section 3.3 System Interface Design | Added narrative to describe what is needed for this section. EIM Enhancement. |

# 1. Overview

The functional design process maps the "what to do" of the Requirements Specification into the "how to do it" of the design specifications. During this stage, the overall structure of the product is defined from a functional viewpoint. The functional design describes the logical system flow, data organization, system inputs and outputs, processing rules, and operational characteristics of the product from the user's point of view. The functional design is **not concerned with the software or hardware** that will support the operation of the product or the physical organization of the data or the programs that will accept the input data, execute the processing rules, and produce the required output.

Suggested Approaches:

**Prototyping** - Prototyping is a user centric design approach which makes it possible for the end users to visualize the system, even before it is being built, and provides a mechanism to validate requirements in earlier stages. Prototypes will also serve to provide screen shots which can be inserted into the Use Case and/or User Interface/Screen Design Documents.

**Use Case Modeling** - A Use Case is a description of a system’s behavior in response to a request that originates from a user. A Use Case describes “who” can do “what” with the system in question and how the various scenarios and errors will be handled.

# 2. System Overview

## 2.1 Context Diagram

This section documents the conceptual level diagram depicting the flow of data between systems. When necessary, decompose system into sub-components when developing sub components.

The Level 0 of the <project name/acronym) Functional Specifications corresponds to the System Context and to the <project name/acronym> System Level (0)

### 2.1.1 <Project Name/Acronym> System Context

The <Project Name/Acronym> System has interfaces with the following external entities:

1. External Entity #1 Name. <include description>
2. External Entity #2 Name. <include description>

The figure below, <Project Name/Acronym> Context Level Data Flow Diagram (DFD) shows the interfaces to these entities. A brief description of the interfaces is provided in the following paragraphs.



Figure 2.1-1 <Project Acronym> Context Level Data Flow Diagram

The interface of <Project Name/Acronym> System with the <External Entity #1> relates to …. <explain>.

*Include one subsection for each external entity describing the interface and the relationships.*

## 2.2 Business Process Flow

This section documents the high-level data specifications. Identify the data that is needed for the application. What data will be consumed from legacy applications? What data needs to be created? What data will flow into which downstream system?

### 2.2.1 <Project Name/Acronym> System Level (1)

The following list summarizes the subsystems that comprise the <Project Name/Acronym> System.

1. <Subsystem #1> - *Include short description of subsystem (1)*
2. <Subsystem #2> - *Include short description of subsystem (2)*

The Figure below <Project Name/Acronym> System Level (1) DFD shows the DFD level 1 of the <Project Name/Acronym> System. A description of each subsystem is provided in the following paragraphs.



Figure 2.2-1 <Project Name/Acronym> System Level (1) DFD

<include detailed description of level (1) DFD>

*The interface to external entities includes processes for the provision of contribution information, either on a regular basis or on demand. It also assists in handling contribution related complaints provision of an ad-hoc query facility to meet the information requirements of other non-automated operations, and reconciliation of individual and <type> records (e.g. comparisons of data from other sources, identification of discrepancies, reporting and correction.)*

### 2.2.1 <Project Name/Acronym> System Level (2)

*<include detailed description of level (2) DFD>*

## 2.3 Dependencies, Limitations, Constraints and Prerequisites

# 3. Functional Design Specifications

## 3.1 Data Specifications

This section documents the data attributes required to satisfy the functional requirements. Required information includes: the data sets required to satisfy the functional requirements, the identification of source systems, the relationship between the systems, legacy applications that provide data, data that needs to be derived from source systems, and downstream (to the application) consumers of the data. To be included in this section is the entity relationship diagram which documents the interaction between the systems that provide the data and also the flow of the data attributes from source to application to downstream systems.

### 3.1.1 Logical Data Model

This section is used to capture the details of the logical data model including the entity relationship diagrams, entity definitions, and attribute definitions. In the absence of a logical data model, document the tasks required to develop one, keeping in mind the following best practices:

* All project participants should be included either as participants or reviewers in the modeling process, including business stakeholders and subject matter experts, requirements analysts, architects, designers, developers, testers, and data modelers and Data Base Administrators (DBA)s. Level of involvement may vary, but the minimum core participants are data modelers, Subject Matter Experts (SME)s, requirements analysts, and architects.
* The Logical Data Model (LDM) is entirely consistent with and at the same level of detail as other project requirements documents.
* The LDM must be directly translatable to English easily understood by business project participants.
* The data model must be understandable to a business audience, whether in graphical/metadata format or expressed as text business rules. If one of these best practices makes the model less understandable to business participants, then the team will maintain a separate business presentation version along with the definitive model.

### 3.1.2 Data Dictionary

Verify if a logical data dictionary exists for data already in production (e.g. data from legacy applications consumed by this application). In the absence of a data dictionary, create work break down structure task to develop one.

### 3.1.3 Data Share Requirements

For data attributes that require access permissions from the source systems, users are required to provide existing Data Share Agreements (if any) or create Data Share Agreements.

Updated data needs are required to be brought to the attention of the Data Stewards. Additional data domains/sources that require additional Data Sharing Agreements will need to be captured in the different application Data Specifications section and additional Data Share Agreements will need to be captured in this section.

## 3.2 Human Interface Design

Insert wireframe diagrams here.

## 3.3 System Interface Design

In this section, document the use of Application Programming Interfaces (APIs), amongst other design considerations, to distribute information downstream. If there is more than one application that received the data, usage of APIs will ensure standardized distribution format for data that flows to multiple downstream systems.

If the project requires the migration of conversion of data, use this section to document the source-to-target mapping, Extraction Transformation and Load (ETL) design specifications, and conversion design. Standards for data conversion must be followed.

## 3.4 Security Structure and Application Roles

In computer systems security, role-based access control is an approach to restricting system access to authorized users. Role-based-access-control (RBAC) is a policy neutral access control mechanism defined around roles and privileges. The components of RBAC such as role-permissions, user-role, and role-role relationships make it simple to perform user assignments.

Three primary rules are defined for RBAC:

1. Role assignment: A subject can exercise a permission only if the subject has selected or been assigned a role.
2. Role authorization: A subject's active role must be authorized for the subject. With rule 1 above, this rule ensures that users can take on only roles for which they are authorized.
3. Permission authorization: A subject can exercise a permission only if the permission is authorized for the subject's active role. With rules 1 and 2, this rule ensures that users can exercise only permissions for which they are authorized.

Additional constraints may be applied as well, and roles can be combined in a [hierarchy](https://en.wikipedia.org/wiki/Hierarchy) where higher-level roles subsume permissions owned by sub-roles.

With the concepts of [role hierarchy](https://en.wikipedia.org/wiki/Role_hierarchy) and constraints, one can control RBAC to create or simulate [lattice-based access control](https://en.wikipedia.org/wiki/Lattice-based_access_control) (LBAC). Thus, RBAC can be considered to be a superset of LBAC.

When defining an RBAC model, the following conventions are useful:

* S = Subject = A person or automated agent
* R = Role = Job function or title which defines an authority level
* P = Permissions = An approval of a mode of access to a resource
* SE = Session = A mapping involving S, R and/or P
* SA = Subject Assignment
* PA = Permission Assignment
* RH = Partially ordered Role Hierarchy. RH can also be written: ≥ (The notation: x ≥ y means that x inherits the permissions of y.)
	+ A subject can have multiple roles
	+ A role can have multiple subjects
	+ A role can have many permissions
	+ A permission can be assigned to many roles
	+ An operation can be assigned to many permissions
	+ A permission can be assigned to many operations

A constraint places a restrictive rule on the potential inheritance of permissions from opposing roles, thus it can be used to achieve appropriate [separation of duties](https://en.wikipedia.org/wiki/Separation_of_duties). For example, the same person should not be allowed to both create a login account and to authorize the account creation.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Subject (organizational role** | **Role****(System Role)** | **Permissions** | **Session** | **Subject Assignment** | **Permission Assignment** | **Role Hierarchy** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

# Approval Information

The signatures relay an understanding of the purpose and content of the document by those endorsing it.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Approve |  | Approve with Modifications |  | Reject |

Comments:

**Approval Signatures**

| Role | Name/Title | Signature | Date |
| --- | --- | --- | --- |
| Client Sponsor |  |  |  |
| DTMB Sponsor |  |  |  |
| Project Manager |  |  |  |

**IMPORTANT!  IN ORDER FOR THE REMAINING PAGES OF THIS DOCUMENT TO FUNCTION PROPERLY, PLEASE DO NOT INSERT/REMOVE ANYTHING PAST THIS POINT!  NOTE:  THIS STATEMENT WILL NOT PRINT, UNLESS PROMPTED.  PLEASE DO NOT REMOVE FROM THE DOCUMENT.**

State of Michigan

Functional Design Document

Instructions

# General Information

**Author(s)**

Record the name of all authors contributing to this document.

**DTMB Authorized by**

Enter the name of the DTMB Client Service Director (CSD) or Information Officer (IO) authorizing DTMB resources to work on the project.

# Privacy Information

# Change Control

This information is to be used to control and track changes made to this system/project document throughout its lifecycle.

# 1. Overview

# 2. System Overview

## 2.1 Business Process Flow

The Business Process Flow should be a high-level process flow with primary focus on the processes that will be supported by the system. It should list organizational functions or business processes that will be served by the system (e.g., payroll, human resources, licensing) and indicate what portion of a process is currently, or is to be, automated and what portion is currently, or is to be, manual.

## 2.2 System Users

This section should identify the potential system users, at a high level. It should also list the departments, agencies, specific groups or other external clients that will use the system, whether at the state, federal or local government level and indicate if the system will be open to the public and the specific groups that are targeted.

## 2.3 Dependencies, Limitations, Constraints and Prerequisites

Dependencies, limitations, constraints and/or prerequisites that could affect the design of the software should be listed in this section (e.g., dependencies on external databases, web services, screen size limitations within a browser, budget and schedule constraints).

# 3. Functional Design Specifications

## 3.1 Data Specifications

### 3.1.1 Logical Data Model

A logical data model should include data specifications sufficient enough to develop physical data specifications, outline data dependencies, relationships, and integrity rules, and specify the attributes for each data element. Additionally, it should identify especially sensitive or critical data and indicate data and transaction volumes.

### 3.1.2 Data Dictionary

A data dictionary is useful for developing context-sensitive help. Initiation of the data dictionary should begin here (in the functional design) and then get finalized in the System Design Stage.

## 3.2 Human Interface Design

A Human Interface Design should:

* Include Use Cases.
* Include the User Interface/Screen Design, unless it is already included in the Use Cases.
* Indicate enterprise, agency or project-specific “user interface standards” that should be complied with (e.g., e-Michigan Web Interface Standards).
* Link the Use Cases to the associated requirements in the Requirements Traceability Matrix.

**NOTE:** If techniques other than Use Cases are used, use this section to describe the User Interface Requirements (e.g., menu hierarchy, data entry screens, search screens, online help, system messages, processing and business rules, algorithms).

## 3.3 System Interface Design

A System Interface Design should specify how the product will interface with other systems. This may include items like the elements of the inputs from and outputs to the interface and the frequency of transmission.

Also, the Use Cases document (SEM-0502) may be tailored to capture System Interface Requirements (e.g., for a Batch Processing interface - a predetermined time schedule could be a trigger; the Pre/Post Conditions are also relevant in case of system interfaces).

## 3.4 Security Structure and Application Roles

This section should identify detailed access requirements and the application roles that will be required for each business process (this can also be captured in the individual Use Case). It should also specify data communication requirements (e.g., SSL, encryption).

# Approval Information