System Design Document

For

UAV Swarm

Team members:

- Samantha Balistreri
- Tyler Wise
- Joseph Moran
- Michael Fornito
- Will Edwards
- Daniela Regueira

Version/Author	Date

TABLE OF CONTENT

SYSTEM DESIGN DOCUMENT

Overview

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

1 INTRODUCTION

1.1 Purpose and Scope

The purpose of this document is to show the architecture and design of the drone swarming simulation constructed with Unreal Engine and AirSim. The scope of this project so far is to develop a swarming algorithm that uses image recognition software to allow drones to swarm in the correct formation.

1.2 Project Executive Summary

This section provides a description of the project from a management perspective and an overview of the framework within which the conceptual system design was prepared.

1.2.1 System Overview

This section describes the system in narrative form using non-technical terms. It should provide a high-level system architecture diagram showing a subsystem breakout of the system, if applicable. The high-level system architecture or subsystem diagrams should, if applicable, show interfaces to external systems. Supply a high-level context diagram for the system and subsystems, if applicable. Refer to the requirements trace ability matrix (RTM) in the Functional Requirements Document (FRD), to identify the allocation of the functional requirements into this design document.

1.2.2 Design Constraints

This section describes any constraints in the system design (reference any trade-off analyses conducted such as resource use versus productivity, or conflicts with other systems) and includes any assumptions made by the project team in developing the system design.

1.2.3 Future Contingencies

A change that might arise that will affect the current design of the system is switching from an image based swarming algorithm to a GPS based algorithm. This would change the design as it would no longer need an image processing software and instead use GPS tracking.

1.3 Document Organization

This section describes the organization of the Systems Design Document.

1.4 Project References

- APAWSAN: Actor Positioning for Aerial Wireless Sensor and Actor Networks by Mustafa Ilhan Akbas and Damla Turgut, University Of Central Florida
- Vision Based Formation Control in Unreal Engine -Air Sim by Kaveh Fatian
- Actor Positioning Based on Molecular Geometry in Aerial Sensor Networks by Mustafa ˙Ilhan Akbas¸ Gurkan Solmaz and Damla Turgut, University of Central Florida

1.5 Glossary

- UAV - Unmanned Aerial Vehicle
- UAS - Unmanned Aircraft Systems
- UE4 - Unreal Engine 4
- MVS - Microsoft Visual Studio
- SBF - Swarm Behavior Framework
- AirSim - An open-source, cross platform simulator for drones, ground vehicles such as cars and various other objects, built on Epic Games' Unreal Engine 4 as a platform for AI research.
- GPS - Global Positioning System

2 SYSTEM ARCHITECTURE

In this section, describe the system and/or subsystem(s) architecture for the project. References to external entities should be minimal, as they will be described in detail in Section 6, External Interfaces.

2.1 System Hardware Architecture

Not applicable for this project

2.2 System Software Architecture

In this section, describe the overall system software and organization. Include a list of software modules (this could include functions, subroutines, or classes), computer languages, and programming computer-aided software engineering tools (with a brief description of the function of each item). Use structured organization diagrams/object-oriented diagrams that show the various segmentation levels down to the lowest level. All features on the diagrams should have reference numbers and names. Include a narrative that expands on and enhances the understanding of the functional breakdown. If appropriate, use subsections to address each module.

2.3 Internal Communications Architecture

In this section, describe the overall communications within the system; for example, LANs, buses, etc. Include the communications architecture(s) being implemented, such as X.25, Token Ring, etc. Provide a diagram depicting the communications path(s) between the system and subsystem modules. If appropriate, use subsections to address each architecture being employed.

Note: The diagrams should map to the FRD context diagrams.

3 HUMAN-MACHINE INTERFACE

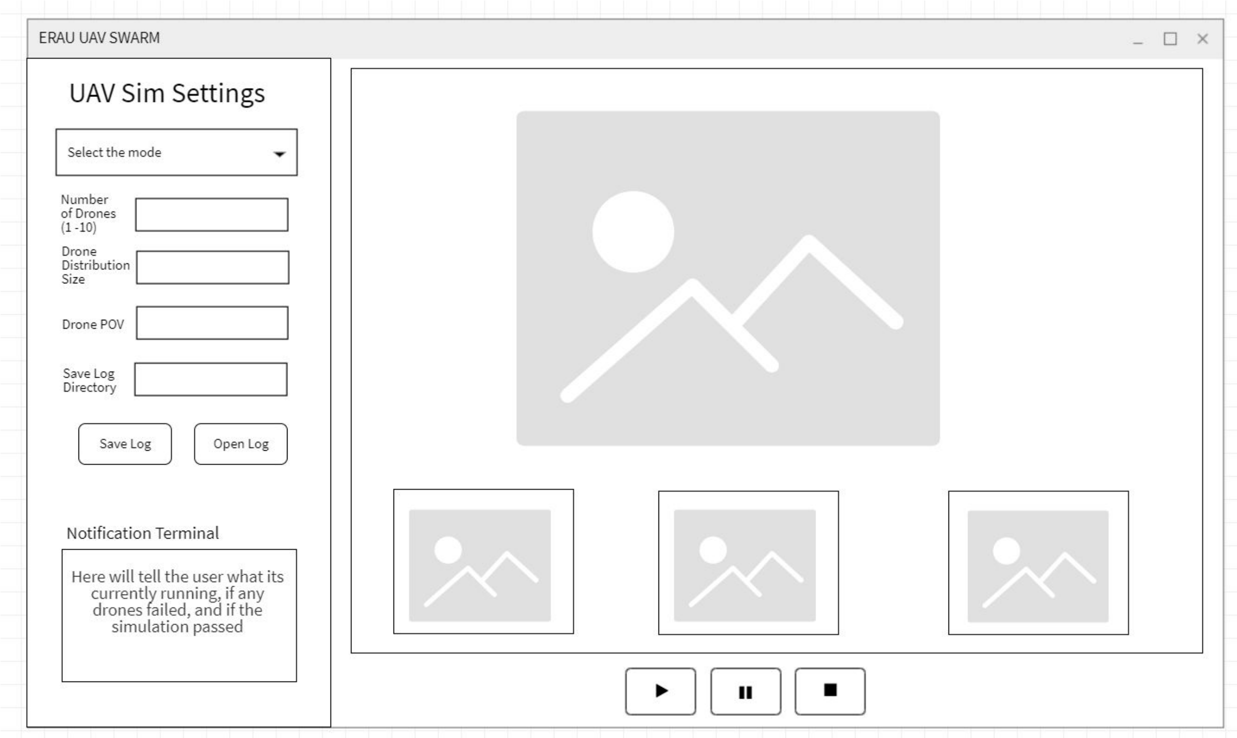
This section provides the detailed design of the system and subsystem inputs and outputs relative to the user/operator.

3.1 Inputs

The diagram below describes input parameters to ensure the user does not face any errors as stated in Section 3.2 and allows the user to freely customize their simulation execution.

Input Parameters

1. **Selection Mode -** The user must select the mode either image processing or GPS location based for the swarm simulation.
2. **Drone Number -** The drone number's range from 1 to 10 for the simulation
3. **Drone Distribution Size -** The drone distribution size is how large the swarm should be for the simulation
4. **Drone POV** - The drone POV is actually an optional feature, but the user may want to view a particular drone if they are curious. The default view is TBD.
5. **Save Log Directory -** The user must enter where to save the log and plot data generated after the execution of the simulation.



3.2 Outputs

The diagram above describes output responses to ensure the user that all inputs are valid, any errors within the system, and to improve the software's fault tolerance.

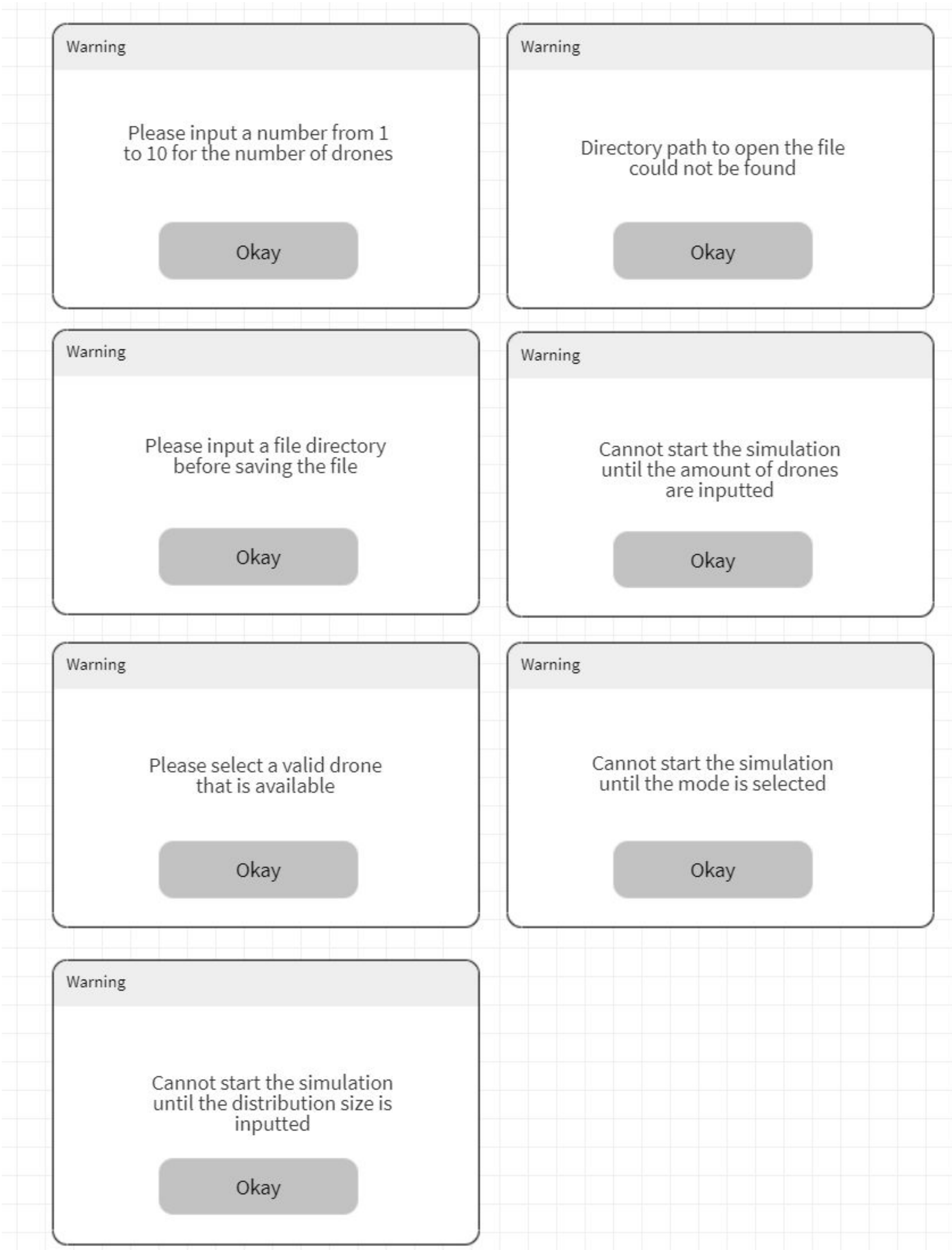
Warning Messages

1. **Drone Number -** The warning is displayed to the user if the user does not properly put the correct range of values as specified in Section 3.1 for the input interface design.
2. **Directory Path For Save -** The warning is displayed to the user if the user attempted to open up the log as described in Section 3.1, meanwhile a directory hasn't been inputted to retrieve the file or the path doesn't exist
3. **Input Directory -** The user attempts to save a file without specifying the save directory path in Section 3.1 for the input interface design.
4. **Failed To Start (Drone #)** - If the user attempted to start the simulation without any drone numbers as specified in Section 3.1 for the input interface design.
5. **POV (Point Of View)** - If the user selected a drone to view, as described in 3.1 for the input interface design, that doesn't exist upon simulation start the warning message will appear.

6. **Failed To Start (Mode) -** A mode for the simulation must be selected, as described in Section 3.1 for the input interface design, error message will appear upon simulation start
7. **Failed To Start (Distribute Size) -** The distribution size for the simulation must be inputted, as described in Section 3.1 for the input interface design, error message will appear upon simulation start

Data Collection

1. **Save Log -** When clicking the save button as described in the diagram located in Section 3.1, the log file and plots will be saved to the destination
2. **Open Log** - When clicking the open log button as described in the diagram located in Section 3.1, the file explorer will open showing the user where his files are located for quick access to view
3. **Notification Terminal** - The terminal will give the user feedback during the execution of the simulation such as: which drones have passed or failed, which mode is running, as described in the diagram located in Section 3.1, save was successful, and current selected settings that are active during the simulation.



4 DETAILED DESIGN

This section provides the information needed for a system development team to actually build and integrate the hardware components, code and integrate the software modules, and interconnect the hardware and software segments into a functional product. Additionally, this section addresses the detailed procedures for combining separate COTS packages into a single system. Every detailed requirement should map back to the FRD, and the mapping should be presented in an update to the RTM and include the RTM as an appendix to this design document.

4.1 Hardware Detailed Design

- Not applicable for this project

4.2 Software Detailed Design

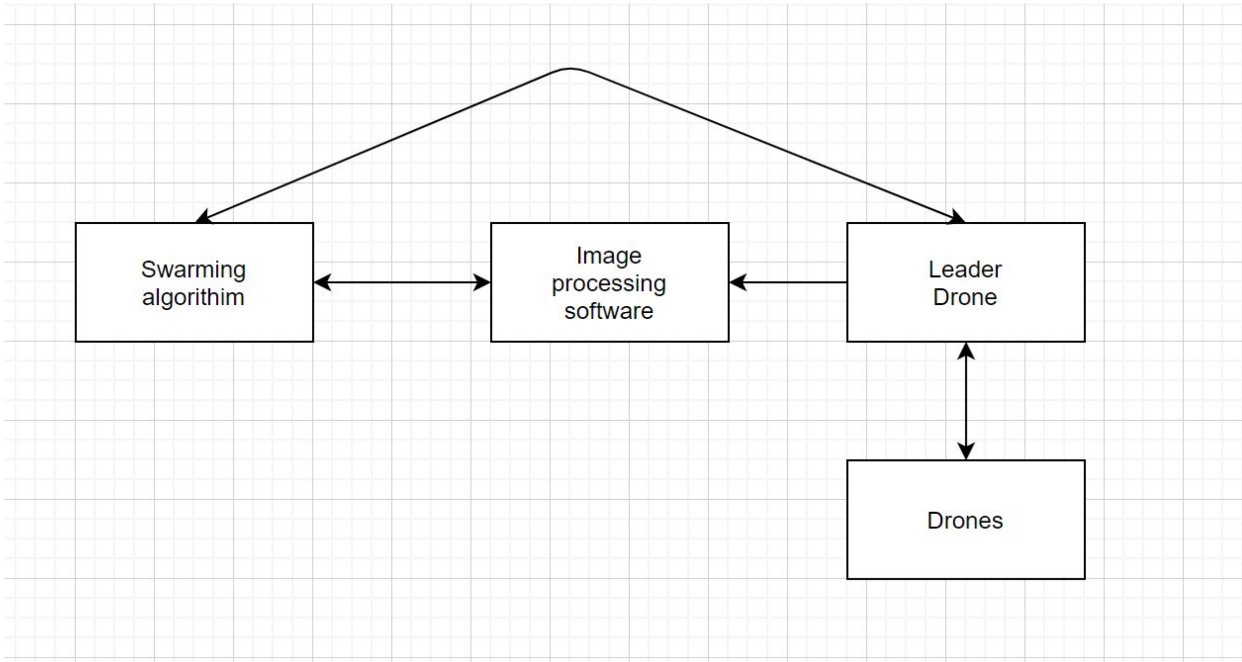
A software module is the lowest level of design granularity in the system. Depending on the software development approach, there may be one or more modules per system. This section should provide enough detailed information about logic and data necessary to completely write source code for all modules in the system (and/or integrate COTS software programs).

If there are many modules or if the module documentation is extensive, place it in an appendix or reference a separate document. Add additional diagrams and information, if necessary, to describe each module, its functionality, and its hierarchy. Industry-standard module specification practices should be followed. Include the following information in the detailed module designs:

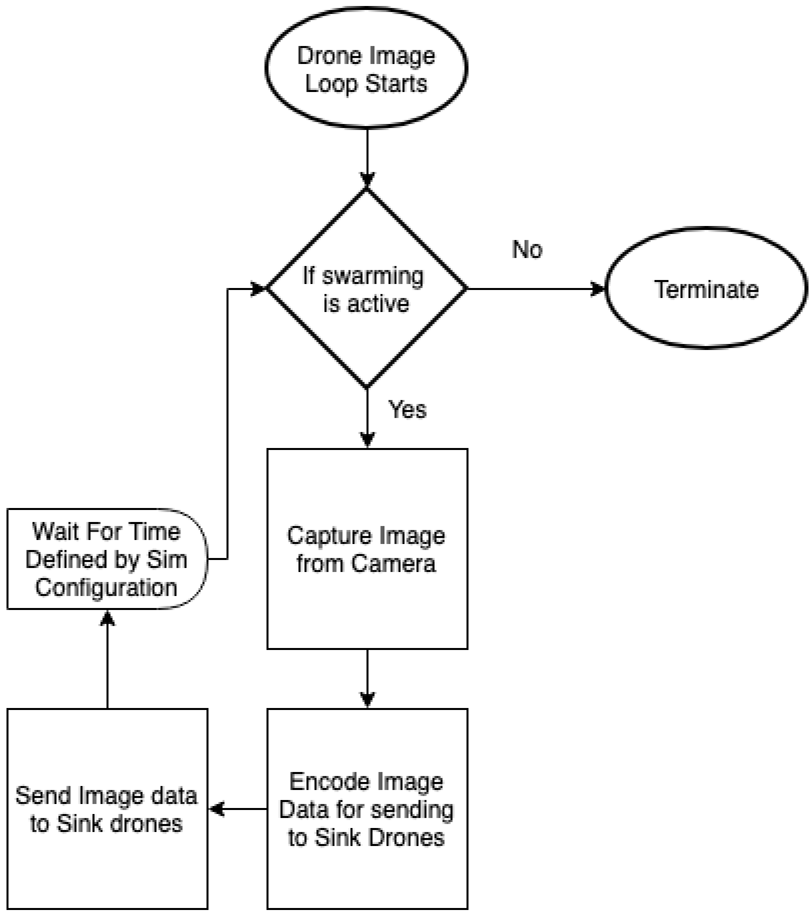
- A narrative description of each module, its function(s), the conditions under which it is used (called or scheduled for execution), its overall processing, logic, interfaces to other modules, interfaces to external systems, security requirements, etc.; explain any algorithms used by the module in detail
- For COTS packages, specify any call routines or bridging programs to integrate the package with the system and/or other COTS packages (for example, Dynamic Link Libraries)
- Data elements, record structures, and file structures associated with module input and output
- Graphical representation of the module processing, logic, flow of control, and algorithms, using an accepted diagramming approach (for example, structure charts, action diagrams, flowcharts, etc.)
- Data entry and data output graphics; define or reference associated data elements; if the project is large and complex or if the detailed module designs will be incorporated into a separate document, then it may be appropriate to repeat the screen information in this section
- Report layout

4.3 Internal Communications Detailed Design

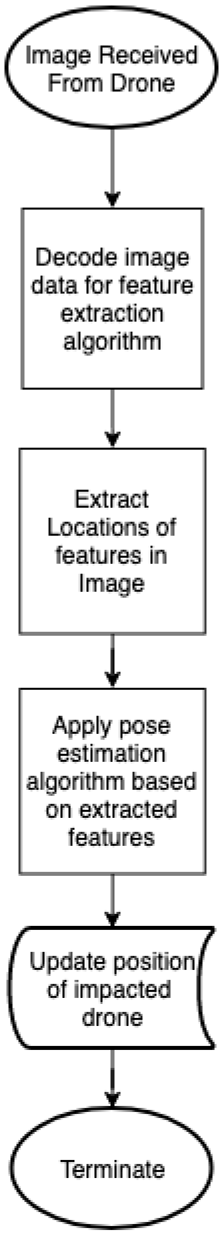
4.3.1Data Exchange Between Drones



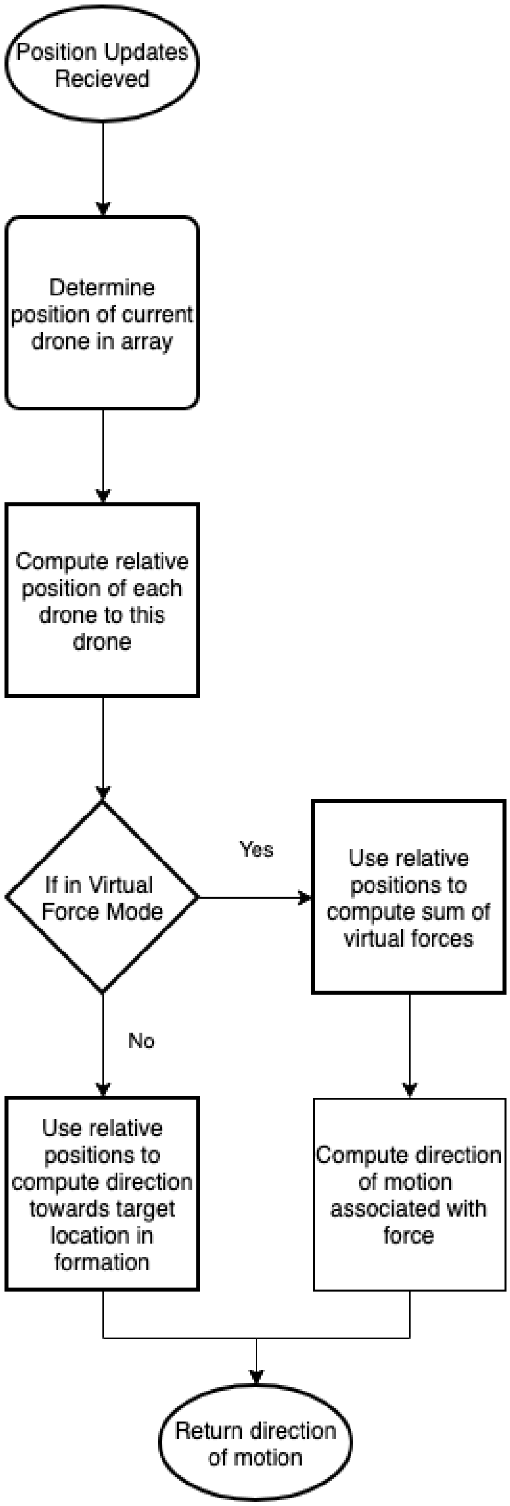
4.3.2Flowchart for Asynchronous Drone Image Capture



4.3.3 Flowchart for Asynchronous Drone Image Processing



4.3.4Flowchart for Asynchronous Drone Position Update Receipt



5 EXTERNAL INTERFACES

External systems are any systems that are not within the scope of the system under development, regardless whether the other systems are managed by the State or another agency. In this section, describe the electronic interface(s) between this system and each of the other systems and/or subsystem(s), emphasizing the point of view of the system being developed.

5.1 Interface Architecture

In this section, describe the interface(s) between the system being developed and other systems; for example, batch transfers, queries, etc. Include the interface architecture(s) being implemented, such as wide area networks, gateways, etc. Provide a diagram depicting the communications path(s) between this system and each of the other systems, which should map to the context diagrams in Section 1.2.1. If appropriate, use subsections to address each interface being implemented.

5.2 Interface Detailed Design

For each system that provides information exchange with the system under development, there is a requirement for rules governing the interface. This section should provide enough detailed information about the interface requirements to correctly format, transmit, and/or receive data across the interface. Include the following information in the detailed design for each interface (as appropriate):

- The data format requirements; if there is a need to reformat data before they are transmitted or after incoming data is received, tools and/or methods for the reformat process should be defined
- Specifications for hand-shaking protocols between the two systems; include the content and format of the information to be included in the hand-shake messages, the timing for exchanging these messages, and the steps to be taken when errors are identified
- Format(s) for error reports exchanged between the systems; should address the disposition of error reports; for example, retained in a file, sent to a printer, flag/alarm sent to the operator, etc.
- Graphical representation of the connectivity between systems, showing the direction of data flow
- Query and response descriptions

If a formal Interface Control Document (ICD) exists for a given interface, the information can be copied, or the ICD can be referenced in this section.

6 SYSTEM INTEGRITY CONTROLS

Sensitive systems use information for which the loss, misuse, modification of, or unauthorized access to that information could affect the conduct of State programs, or the privacy to which individuals are entitled.

Developers of sensitive State systems are required to develop specifications for the following minimum levels of control:

- Internal security to restrict access of critical data items to only those access types required by users
- Audit procedures to meet control, reporting, and retention period requirements for operational and management reports
- Application audit trails to dynamically audit retrieval access to designated critical data
- Standard Tables to be used or requested for validating data fields
- Verification processes for additions, deletions, or updates of critical data

Ability to identify all audit information by user identification, network terminal identification, date, time, and data accessed or changed.