*Architecture Document for*

Smart and Connected Health Alert System (SCHAS)

Version: 1.5

Last Updated: 01/10/2014 Table of Contents

1 Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms and Abbreviations 1

1.4 References 1

2 Architectural Goals and Constraints 1

2.1 Goals 1

2.2 Constraints 2

3 System Context 2

4 Business Process View 2

5 Use Case View 2

5.1 Main Actors 4

6 Logical Architecture View 4

7 Deployment Architecture View 5

8 Data Architecture View 5

9 System Process View 5

10 Implementation Architecture View 5

10.1 Database Design 6

11 Component Architecture View 6

12 Architecture Road Map 6

13 Architecturally Significant Risks 7

14 Steel Threads 8

15 Architectural Open Issues 8

# Introduction

## Purpose

The purpose of this System Architecture Document (SAD) is to provide an overview of the ICT project, brief overview of the architectural components and implementation details. This document should be considered as a vision for system future state and all decisions are captured. This document contents will change, as implementation needs to be calibrated.

## Scope

This document can be used by:

* System implementers: as implementation guideline.
* Project Managers: To create roadmap of work and to ensure implementation progress.
* System owners: To understand the system boundaries.
* System Architects: To evolve the system as it is built and to integrate to other external systems.

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| Triggers | Symptoms that cause an attack |
| PFM | Peak Flow Meter |
| SCH | Smart and Connected Health |
| Inhaler | A device used to increase the airflow |
| SCHAS | Smart and connected health alert system. |

## References

|  |  |
| --- | --- |
| ICT Fund Document | See ICT Fund document |

# Architectural Goals and Constraints

## Goals

This section provides goals in terms of architectural qualities that must be exhibited by the SCHAS system.

The top three qualities of the system must be:

* Simplicity: the system must be Simple
  + Simplicity includes the system will choose to implement components simplicity and readability over the efficiency.
  + It must be easily understood and provide ability to add new components easily
  + System uses simple frameworks including any third party tools
  + Data is managed in a way that allows it to be used for many purposes
  + The system uses open source frameworks such as Linux O.S. and simple file systems
* Exploratory: The system allows for exploratory activities such as visualization of location, movements, overlay of environmental factors etc.
  + This includes ability view data in multiple ways (tabular, graphical, spatial etc.)
  + This includes customizing the views (changing layers in GIS)
* The system is *Maintainable* – by that it means, it allows quick detection of any errors via tools and techniques to trouble shoot the system or subsystems.
  + This includes, ability to move data/software from one system to another system
  + The system includes just enough documentation to deploy and manage the working of it.

## Constraints

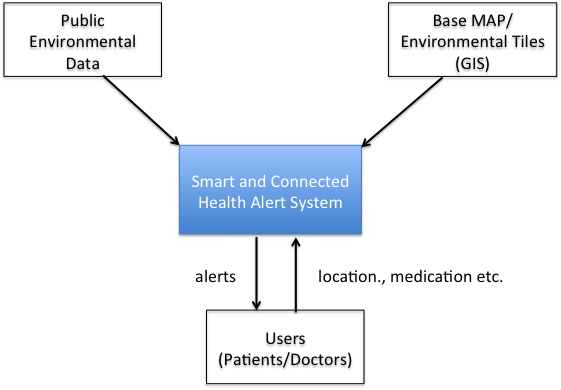
This section describes the constraints that application design such as: design and implementation strategy, development tools, technology, team structure, schedule, legacy code etc.

As such SCHAS will not have any legal constraints other than ensuring that the collected data does not have any personal identification. SCHAS also does not foresee any compliance to any standards.

# System Context

The following diagram shows SCHAS in the context diagram. The system will have minimum external interfaces and dependencies. The system will use base maps from freely available resources such as openstreetmap or base map from open layers. It will use freely available Web-Mapping-Services (WMS) to get environmental tiles to display on the map. Environment data for relevant locations will be downloaded from any relevant available resources.

The system will input users location information. The system will consume medication and voluntary inputs from patients; it will also have a component to send alerts to users mobile devices.



# Business Process View

NOT Applicable for the project. This section will be updated after the initial phase.

# Use Case View

The following diagram shows the architecturally significant use cases.

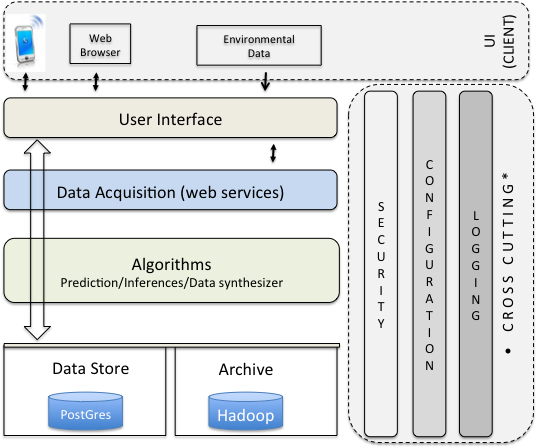
Three main functional use cases are identified.

| **Significant Use Cases** | **Description** | **Primary Actor(s)** | **Use Case Goals** |
| --- | --- | --- | --- |
| Store Patients movements and volunteer inputs | The goal of this use case is to capture the location information and other related information and send it to server successfully. Upon receipt Server will store the information in the database.  The patients mobile devices can capture the data from  (1) Upon usage of peak flow meter  (2) Upon usage of inhaler  (3) Patients input when attack occurred | * Patients sends data * System will receive and store it in the data store | * Send location data to server * Receive data and store it in server * Ensure no data loss |
| Architecturally Significant Requirements:   * Data received is stored securely and with minimum loss * Data received will clearly identify the sender and ensure that it is valid * Performance - Allow simultaneous sends and no data corruption * Make data available for analysis and online viewing | | |
| Ingest Environmental Data | Environmental Data (this include discrete and weather map tiles) will be stored for relevant time period. | * System Admin | * Environmental Data captured * Data is stored in the local storage * Data is reviewed and cleansed * Data is available for use |
| Architecturally Significant Requirements:   * Identify relevant time period and download environmental data * Ensure data is clean | | |
| Allow selection of interesting data to user | Allow users to review the data and select interesting data | * Researcher | * Selecting exploring interesting data |
| Architecturally Significant Requirements:   * Determine the data access efficiently * Usability of the system | | |
| Display patients stored data on the web browser | Process for defining projects, collection of expenditures and posting to the general ledger | * Researcher | * View on web * A way to validate the environmental data * A way to validate the customer inputs |
| Architecturally Significant Requirements:   * A reliable, secure interface is provided for interactive view and time reporting systems to process time transactions. * A reliable, way to explore the data visually | | |

## Main Actors

* System Administrator  
  Manages the system, adds users and loads data from external systems
* Researcher  
  Conducts exploratory analysis using the data collected from the patients and using the meta data such as environmental data. This role includes doctors and nurses as well.
* Patients  
  those who will volunteer to send the data required for the analysis.

# Logical Architecture View



The diagram above shows the logical overview of the system. It mainly consists of user interface layer to interface with mobile devices and web browsers.

User interface (UI) layer will be responsible for showing maps, of locations, overlaying environmental tiles (if available), allows users to load and view data for patients with interesting data.

UI layer will also implement minimum interfaces to manage users including patients. UI will be implemented using MVC pattern in most cases. Some exploratory analysis will be developed using

Data Acquisition system will be responsible for collecting the data or providing data to UI. This will be mostly implemented using web-services.

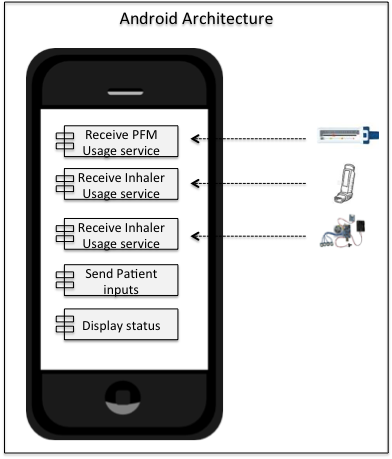
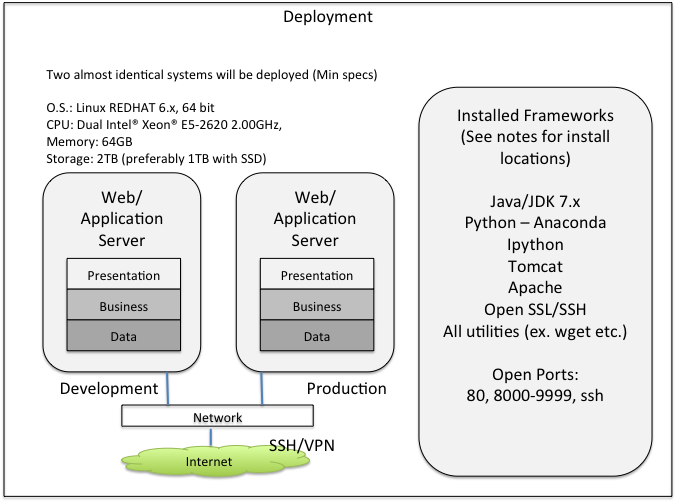
Algorithms will implement various machine learning algorithms, prediction services.

Data storage will be responsible for storing the data. The system will use RDBMS using Postgres and PostGIS database to store and manage data. RDBMS will have data to allow efficient querying; the system will use Hadoop system to achieve data.

The system will implement minimal security to ensure the data and view are accessible to valid users.

# Deployment Architecture View

The system deployment is as shown below.



# Data Architecture View

The significant data elements of the system are as follows:   
All these data are time stamped in GMT

* Patients movement data
* Patients medication and inhaler data
* Patients data that indicate when an attack occurred
* Environmental data

# System Process View

Most system processes will be implemented as state-less. There will not be a need for long standing transactions. However there are components that have interdependency on other components and thus must be available to ensure that overall working of the system. Such dependencies are shown below.

Web services are the central part of the overall system. Web services depend on the tomcat application server.

|  |
| --- |
|  |

# Implementation Architecture View

The implementation uses the following technologies.

* Linux Redhat 6.x (or Ubuntu) version of O.S
* Tomcat as application server front ended by Apache server
* Hadoop file system will used in the second phase of the project to store large amounts of data. In the initial phase all files will be stored locally
* Postgres with PostGIS extensions will be used as database.
* The project will use Java, Python as main programming languages.
* JUNIT for writing test scripts
* Openlayers will be used for displaying the GIS maps
* Javascript will be used for most of the client facing systems
* Jlog for logging

## Database Design

The following simple schema will be implemented on the Postgres +PostGIS DB.



This is the simple database schema that will be created to store the data for the initial prototype.

User table will be used to stored users who are registered with their id. Each user will be assigned a mobile ID that they would use to report the readings.

Env table will store the environmental data. The columns “values” in this table will be used to hold any key value pairs stored as JSON formatted strings.

Health table will store the users input collected from using PFM, inhalers and voluntary inputs of attacks or any medication intakes.

# Component Architecture View

# Architecture Road Map

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Comments** |
| System setup and Environment setup | Set up servers and create development environment |  |
| Develop initial web services and establish connectivity of mobile devices with data collection service | Initial prototype to collect health and environmental data |  |
|  |  |  |
| **Android** |  |  |
| Application on Android devices | Application interface installed on Android devices to collect and send data to server | Android application download and installed from Google Play store |
| PMF sensor design and integration with Android phone | Develop PMF sensor and test the connectivity with the Android device |  |
| Inhaler sensor design and integration with Android device | Same as PMF device integration |  |
| Component to send location information from Android device | Receive location information from the devices along with user input data |  |
|  |  |  |
| **Data** |  |  |
| Environmental Data ingest | Investigate environmental data from openweathermap.  Ingest samples of vector data and raster data. | Create services/algorithms to get the weather for a given location and time. |
|  |  |  |
| **User Interface** |  |  |
| Openlayers to show map | Display base map and allow showing of trajectories of the patient (s) for a time period.  Show the indications of the attack or medication intakes. This will allow users to choose the patients whose data will be displayed. |  |
| CRUDX Users | Manage users for the system (create, delete etc.) |  |
| Users exploration | Allow exploring users and their corresponding available data in order to find interesting users |  |
| Graphs | Show charts and graphs of users movement and corresponding environmental data |  |
| Display Environmental data | Show various environmental data using charts and maps. Display weather data on the map using weather tiles |  |
|  |  |  |
| **Algorithms** |  |  |
| Interpolate environment | Determine the weather by interpolating the data between location and time |  |
| Predict | Predict attacks based on the users data |  |

# Architecturally Significant Risks

Risk: Manage the data, data archiving, and security of data

Mitigation: Consistently archive the data and move it into secured place

Risk: Availability of the system and system administration issues:

Mitigation: Handled manually with hard reboots; eventually move to Amazon cloud services

# Steel Threads

## Users movement information

Collect users location information along with data about environment or any voluntary inputs and send to data collection service

## View GIS maps that overlays user’s information

Select interesting users for whom the data has been collected and display them by overlaying it on a GIS map that shows a map along with weather conditions at the corresponding time

## Predict and send alerts to users/doctors

Create prediction service and send alerts to users when the system predicts a attack could happen.

# Architectural Open Issues

|  |  |  |
| --- | --- | --- |
| **Issue** | **Description** | **Resolution** |
| None Identified |  |  |