Functional Requirements Document
Sensor Logging Array

Codename: senseme

Version	Description of Change	Author	Date
1	Initial creation	Andrew Koerner	10/31/2012

CONTENTS

1. INTRODUCTION

There is an inherent need to be able to effectively track simple state changes of company resources over time. This need can be applied to any physical real world object that has a series of finite states. The prototyping effort to generate a simple system to track state changes in company resources will be referred to as *senseme*.

1.1 Purpose

The purpose of this Functional Requirements Document is to outline and provide an understanding between the group(Andrew Koerner) and users of these tools. The careful documentation of the use and requirements for the software/hardware system will allow mutual understand of the project from a development and usage point of view.

1.2 Scope

The scope of this document is to outline functional requirements for the prototyping effort of the *senseme* project.

1.3 Background

In the business world there are many finite state resources that need to be managed on a day to day basis. Some examples being doors, cash drawers, garage doors, locks, lights, appliances et cetera. It is interesting and necessary to be able to track physical states of objects over time.

1.4 References

- This project will be consuming proven technologies such as HTTP 1.1, TCP/IP, ResTful web services and database driver applications. Each Technology can be independently referenced.

1.5 Assumptions and Constraints

Assumptions:

- This system will be sufficiently scalable to meet the needs of the GameRoom in its current state.
- A hosted server or in house physical server will be necessary for this system.
- This system will meet all the requirements outlined within the User Stories document

Constraints:

An entirely new software/hardware system is being created using existing and proven platforms. The first iteration cycle of this project will conclude the first week of December. 2012. The following also play a role into project restriction:

- This prototype system will NOT support analog sensors in the first iteration.
- This is a prototype effort and is not meant to be a full production effort
- Do to the nature of a prototype effort initial budget and time expenditure will be limited.

1.6 **Document Overview**

The remainder of this document covers the methodology used to elicit requirements and compose this document, the functional requirements of the system to be implemented and other requirements of the *Senseme* system. First the methodology of requirement elicitation will be discussed. Next, the document discusses the function requirements of the system. Finally, all other requirements are discussed.

2. METHODOLOGY

The approach of this project will be two fold, firstly to use a proven embedded development platform to talk to a server with the aim to persist data. Secondly, provide useful reports and notifications to users of the system.

3. FUNCTIONAL REQUIREMENTS

3.1 Context

Senseme is divided into two separate systems that will be created and coupled meet the necessary requirements. The first is the embedded system platform that will monitor state changes in finite state sensors. This system will then post state changes “events” to an external server where the event will be logged. The final element to this system is the server stack. The server stack will be responsible for receiving the events from the embedded systems and posting them to a database. The server stack will also include reporting of event data and basic system configuration as well as intelligent notification using aggregated event data. The server stack will support many devices of which have many sensors.

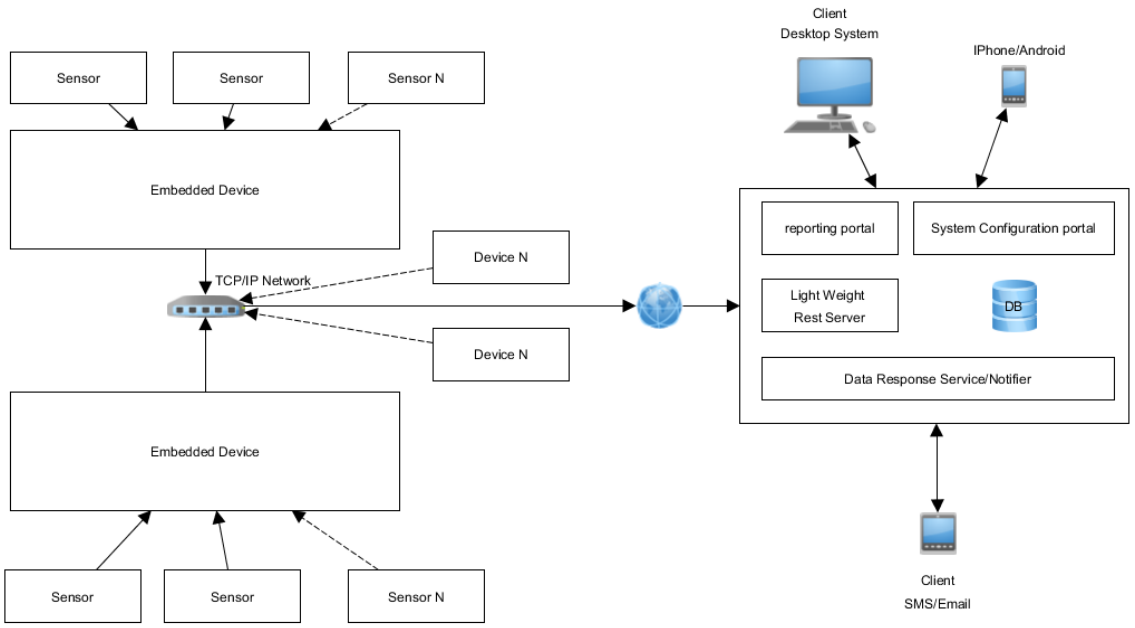

Figure 1. Context Block Diagram for the *Senseame* system.

3.2 Logical Data Model/Data Dictionary

Event – An encapsulation of any state change that may occur to a sensor

3.3 Functional Requirements

Tool 1

3.3.1 Interface

- Web based reports portal
- Web based configuration portal
- Standard email
- SMS

3.3.2 Input Data

- Events generated by the embedded platform

3.3.3 Data output

- Reports generated from event data

4. OTHER REQUIREMENTS

In addition to the functional requirements outlined in the previous section, the additional requirements of the system are outline in the following sections.

4.1 Hardware Interfaces

Senseme will require the integration of physical sensors to track state changes in real world objects. The physical sensors will be coupled with an embedded device to interface with the software system.

4.2 Software Interfaces

Senseme will require a server software stack including: Rest client, data response service/ daemon, reporting interface/portal and configuration portal. The required embedded systems that will be used will require persistent logging, persistent configuration and lightweight http stack.

4.3 Communications Interfaces

Senseme will consume the TCP/IP and HTTP1.1 software stack to transmit to the server and using Restful communication as the data transmission schema. The server portion of *Senseme* will consume the same communication protocol to communicate with clients.

4.4 Maintainability

The system will require knowledge of server administration, basic networking and hardware repair practices to maintain the integrity of physical sensors.

4.5 Hardware/Software Requirements

The system requires an embedded system with an Ethernet adapter. Each embedded system can support up to 16 sensors without extended development or added circuitry. A hosted server capable of running a database platform and scripting language will be necessary. The server can be hosted on site or externally.

4.6 Operational Requirements

This system must reliably caching and persist event data in order to be retrieved and used to trigger notifications to users.

4.7 Security and Privacy

The embedded platform being used in this system has limited computing power. This limits the possible security schemas that could be implemented. The application of symmetric key cryptography for authentication and data transmission will be used.

AES256 will be the algorithm used to accomplish this. This will require a fixed shared secret between the server and devices.

4.8 Reliability

The effective use of the system and purposes of the system require a high degree of reliability. If the system experienced downtime during a critical moment then the entire system will be rendered effectively useless.

4.9 Recoverability

Senseme does not retain any data that has long term significance. Recoverability is not a high priority.

4.10 System Availability

The system should have an extremely high level of availability provided no sensor malfunctions. Data transmission will be asynchronous and thus can be caching if internet connectivity is down.

4.11 General Performance

A. Any report queries are expected to generate a report within in 10 seconds.

B. Any high priority state changes are expected to propagate to a user in no more than a minute.

C. High resolution state changes are expected to be supported i.e., frequent changes in sensor states.

4.12 Capacity

This system will be designed to be scalable. The prototype effort will support many sensor arrays with many sensors in multiple brick and mortar locations.

4.13 Data Retention

- Events will be passed and stored on an external server. Events will also be retained in a flat log file within each embedded sensor array and thus can be latter retrieved manually regardless of server health.
- Configuration settings will also be retained on the server. Basic device specific configuration will be stored locally within the sensor array.

4.14 Error Handling

When an error occurs at the embedded system level logs will be kept in flat files stored on an SD card on board each instance of the embedded system. Logs will be kept in flat files if any errors occur within the server stack also.

4.15 Conventions/Standards

This system will inherit all the conventions and standards outlined by the technologies applied to complete the system.

5. APPENDIX A - GLOSSARY

Senseme: The entire software and hardware system developed over the course of this project.

SMS: Short Message Service.

Event: Persistent record of a sensor state change.