ProVaxx™ Vaccine Monitoring System

System Design & Modeling

Version 1.0

11/17/2016

Aaron Baker, Andy Rusinek, Daniel Schaeffer, Minh-Nhut Dang

REVISION HISTORY		
REVISION	DESCRIPTION	DATE
1.0	INITIAL RELEASE	11/17/2016

TABLE OF CONTENTS

1 PROJECT OVERVIEW

1.1 Assumptions:

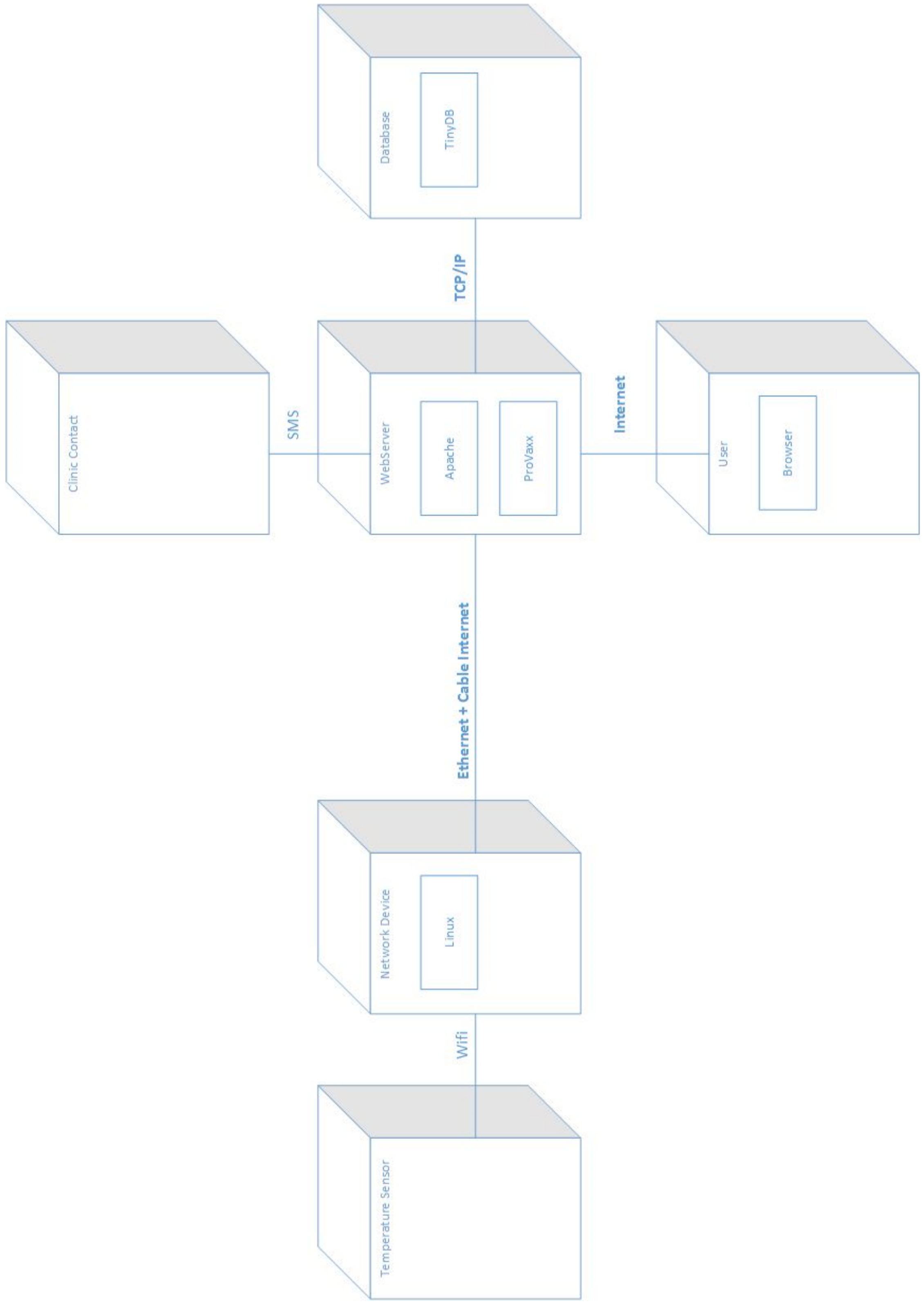
- Each temperature sensor has an internally programmed ID
- Each temperature sensor has a programmed temperature threshold condition (triggers notification to the specified contact)
- A clinic may have multiple temp sensors monitored by a single networking device

1.2 Design Decisions

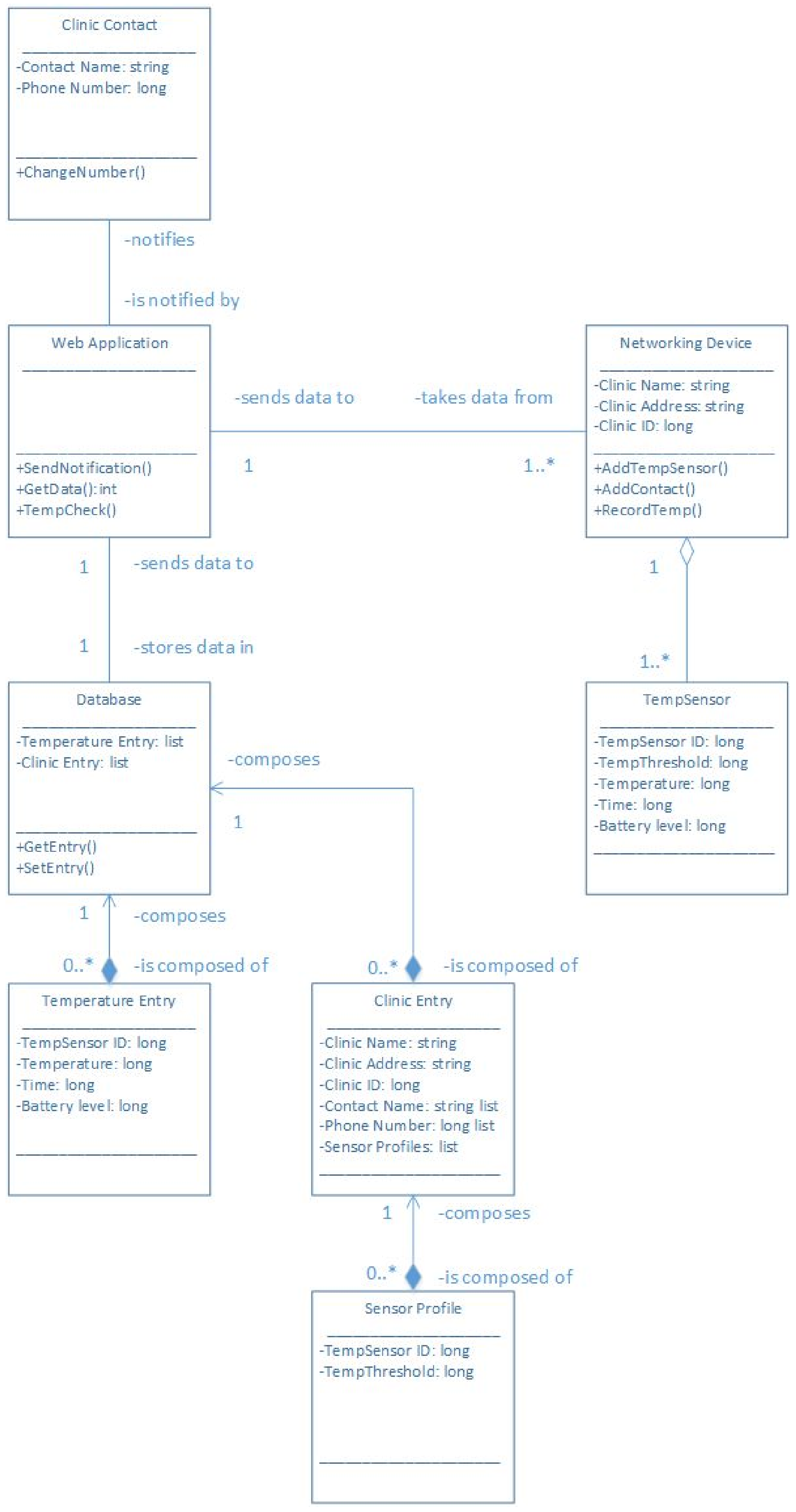
- Each networking device is programmed with clinic info and contact info
- ProVaxx™ is the project’s own application for handling temperature data and notifying clinic contacts
- Battery Powered Temperature Sensor
- Temperature sensors are connected via wifi to a networking device

2 DESIGN & SPECIFICATION DOCUMENTS

2.1 UML PHYSICAL VIEW



2.2 UML CLASS DIAGRAM

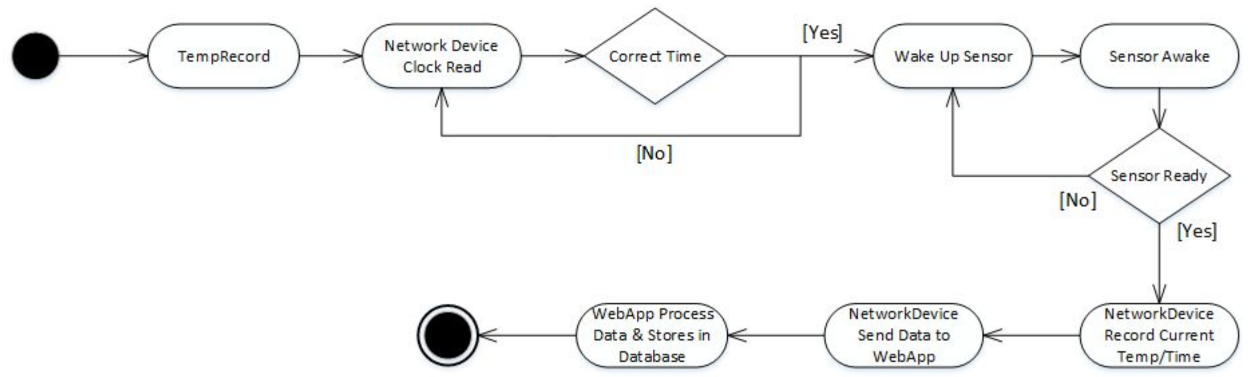


2.3 UML USE CASE DESCRIPTION

USE CASE DESCRIPTION	
Use Case	AddTempSensor
Description	This use-case occurs when a new vaccine temperature sensor is added to the vaccine monitoring system. The system prompts the user for manual verification of accurate temperature readings. Upon successful verification, the system records the current time and date to display when the last temperature sensor was installed.
Actors	User, Network Device, ProVaxx™
Assumptions	Sensor has charge, Sensor is ON, Sensor is not removed during syncing process, Network Device is on and set up, and user has existing ProVaxx™ profile for Network Device.
Steps	1. Temperature sensor is turned on by the user. 2. Temperature sensor begins transmitting and is detected by Networking Device, contacts ProVaxx™ to modify database. 3. User navigates in browser to ProVaxx™ Network Device setup page and is prompted to begin sensor setup 4. Upon clicking 'Yes', user will be brought to screen displaying sensor's current readings, sensor ID, and current battery level. 5. Browser then prompts user to do a manual verification of the current temperature in the freezer and correct sensor ID (displayed on label). 6. Verifies that user has checked

	temperature's current readings for accuracy via a confirmation dialogue. 7. Prompts user for temperature threshold for notification 8. Upon pressing submittal, display message stating successful calibration and successful addition of temperature sensor. 9. The system then records the current time and date and stores it on the database along with the verified temperature sensor ID.
Variations	If user presses 'Not Now' on initial sensor setup, user is redirected back to the previous page on the Browser and is prompted again when the user navigates to the ProVaxx page.
Non-Functional	**Priority**: Must be completed before TempRecord can be executed.
Issues	

2.4 UML ACTIVITY DIAGRAM


TempRecord
Network Device Clock Read
Correct Time
[Yes]
[No]
Wake Up Sensor
Sensor Awake
Sensor Ready
[No]
[Yes]
NetworkDevice Record Current Temp/Time
NetworkDevice Send Data to WebApp
WebApp Process Data & Stores in Database

2.5 HTTP GET SPECIFICATIONS

An easy format for passing agnostic string data entries is CSV (comma-separated values) which be easily parsed by string parsers in many languages. The data that needs to be transmitted is:

- Sensor ID
- Time stamp
- Temperature reading
- Battery voltage

This hierarchical order also makes some intuitive sense (which device>when>relevant data).

Data format:

- Sensor ID: A unique integer value that we can assign. A string of digits.
- Time stamp: Because it provides a consistent convention with many string parsing libraries available it makes sense to use a representation such as ISO8601 UTC time codes: [YYYY]-[MM]-[DD]T[HH]:[MM]:[SS]Z
- Temperature reading: Floating point value in Celsius. Good string parsing libraries can parse arbitrary, signed floating point numbers so it can be represented as a string of digits with a decimal point.
- Battery voltage: Floating point value in V of the battery voltage level. We can also use string floating point numbers for the same reason as temperature.

So our HTTP GET return string should read:
"[ID NUMBER],[UTC TIME STAMP],[FLOATING POINT TEMP IN C],[FLOATING POINT BATT VOLTAGE IN V]"

We can also stipulate that the specification is indifferent to whitespace and case because string parsers can strip it easily.

Example: "100251,2016-12-10T15:45:20Z,-0.51,3.8125"