Remote Home

Requirements and Research Document (Concept of Operations)

Prepared By:
Christopher Jensen
Joshua Kinkade
James Wiegand

Revision History

Date	***Author***	***Version***	***Comments***
12/13/12	*Christopher Jensen*	*1.0.0*	*Adapted from James' initial draft*

Table of Contents

1.0 Overview

This document describes the Remote Home system in detail. The Remote Home is a system for deploying and controlling household devices, such as garage doors and sprinkler systems, from an iPhone application. The overall system includes a gateway for end users, an iPhone app and API for developers, and a set of hardware products which allow for control of the devices.

1.1 Scope

This document contains stakeholder information, initial user stories, requirements, proof of concept results, and various research task results. For additional information, please reference the Software Design Document, SDD.pdf, in the Documents directory of this project.

1.2 Purpose of the System

The Remote Home system is designed to provide vendors with a platform from which any number of devices can be deployed. It is light-weight, scales easily and can be deployed by end-users with no more effort than wireless routers, printers or similar computing equipment.

2.0 Stakeholder Information

From a pragmatic standpoint, the individuals with the greatest stake in this project's successful completion are the student developers, who will rely on this project to graduate. L3 Communications is sponsoring this project, and thus also benefits from successful completion. The platform, if implemented well, will enable developers of new products to more easily integrate their products with internet applications. Present manufacturers of home appliances and other goods would also benefit from having a well-documented and solid framework for deploying applications to control their products remotely.

This project will eclipse prior remote appliance management systems, and could reasonably be adapted to include such devices as part of the specification if time is available, allowing users of previous systems to retain their existing products while implementing them through our service, allowing us to avoid any negative consequences for earlier adopters.

2.1 Customer or End User (Product Owner)

June Knight, with L3 Communications, is the product owner. Although she will not be directly acting as point of contact for the team, she will be making her requirements and approval known through the Scrum Master, Dr. McGough. She is also free to contact us directly at her discression.

2.2 Management or Instructor (Scrum Master)

Dr. Jeff McGough will be acting as Scrum Master for this project. He will be responsible for ensuring group dynamics and facilitating dialog. Although he will be present at major meetings, his responsibilities will not necessarily involve directing them himself; such a task shall be handled by whoever feels sufficiently confident that day.

2.3 Investors

L3 Communications has invested in this project, and will be represented by June Knight through her contact with Dr. McGough.

2.4 Developers | Testers

The developers for Remote Home are Joshua Kinkade, Christopher Jensen and James Wiegand. Brian Vogel will be joining the team during the second semester. All further details can be found in the Software Development Document.

3.0 Business Need

Smart phones and tablet computers are the most rapidly expanding market on the planet. The sudden rise of 3G and 4G “always-connected” devices has meant an increase in the expected availability of information, and there is nothing people are more likely to be concerned with than the status of the very appliances they interact with on a day-to-day basis.

Concerns about “Did I leave the garage door open?” and “Did I leave the plants' water on?” have plagued adults for years, and almost all children have been locked out of their house at some point or another for at least a few hours. Clearly, there is a substantial need for people of all ages to have access to basic household appliance status, such as garage door or sprinkler system.

Because of the recent rise in mobile computing, status updates about devices have never been more plausible. Although it was previously expensive to have and maintain a system for getting status about devices, modern technology can integrate most household appliances into systems which can connect to a household internet connection and a personal computing device to provide real-time updates on different devices.

For this to happen, a platform must exist which facilitates communication between appliances and the end user's computing platform. We shall create this interface for communicating device status and controlling the device in turn, allowing us a clear advantage in the marketplace and enabling users of our platform to create devices with greater ease and reliability than if they had to implement using low-level libraries.

4.0 Requirements and Design Constraints

The overall software design will match that specified in the Software Design Document.

4.1 Project Management Methodology

The stakeholders might restrict how the project implementation will be managed. There may be constraints on when design meetings will take place. There might be restrictions on how often progress reports need to be provided and to whom.

What system will be used to keep track of the backlogs and sprint status?

Will all parties have access to the Sprint and Product Backlogs?

How many Sprints will encompass this particular project?

How long are the Sprint Cycles?

Are there restrictions on source control?

5.0 User Stories

This section can really be seen as the guts of the document. This section should be the result of discussions with the stakeholders with regard to the actual functional requirements of the software. It is the user stories that will be used in the work breakdown structure to build tasks to fill the product backlog for implementation through the sprints.

This section should contain sub-sections to define and potentially provide a breakdown of larger user stories into smaller user stories.

5.1 User Story #1

When the user first starts up the app, or when a user decides to add a device, the add device screen will load. This screen will have two text input boxes. The first box will have a field for the device identification (DID), the second box will have a field for the device name. The DID will be located on each device, the name is a name the user specifies for her/his own use.

5.2 User Story #2

When the user opens the app and there is data in the SQLite database, the user will be presented with a list of devices. Each device will be under a descriptive class designation. For example, if the user has a garage door opener called "Right Garage Door Opener" it would be placed under the "Garage Door" header. Each cell will have a green or red dot to indicate the status of that device, with green corresponding to online and red to offline. If a user touches a device that is online, it will open the appropriate controller view. If the user touches an offline object, the user will be presented with a screen that gives an error message and a human-readable description of what may be going wrong.

5.2.1 User Story #2 Breakdown

If there is at least one device registered, the system can show all registered devices to the user. If there isn't at least one device registered, there is nothing to show the user. All of the devices are grouped in a hierarchy, i.e. each base station has its own group of devices, and each device attached to a base station belongs to some category. The example used is garage doors, but it could just as easily be lights or sprinklers. The hierarchy allows the user to quickly find exactly what to turn on or off without having to scroll through a list of names that someone else may have picked out at random, making finding the right button faster and easier.

Because an offline device cannot be operated, a simple error screen is generated instead. This screen will help users with contacting technical support if necessary, and provides a more graceful handler than simply having a pop-up message or other transient dialog.

5.3 User Story #3

The status view will appear when the user clicks on a device from the main view that is listed as offline, or when the user selects status from the navigation bar on a controller view. The status view will tell users the status of the device (online, offline) at the top of the view. If the device is offline, a code will be presented in the middle of the view; this code will be for technical use. For example if the user has a garage door opener called "Right Garage Door Opener" that is listed offline, because the server cannot communicate with the device the code ERR:001 could be displayed. At the bottom of the view there will be a text box that has an English description of the problem and a potential remedy to said problem. In the above example, "The control station cannot communicate with the device, please ensure that the device is turned on and connected the the control station." could be displayed.

5.3.1 User Story #3 Breakdown

Devices are not always online. Sometimes a circuit breaker gets thrown or a fuse blown, or something's batteries wear out. Sometimes a device has been mis-configured, or has hardware failure. To make troubleshooting easier, the base station will attempt some basic diagnostics (Is the device no longer responding to the base station? Did it start complaining about something? Did the device get moved and is now barely within range?). Exact codes will probably be dependent on time more than anything, but will cover at least “Device is no longer responding” and “Device is actively reporting an error.”

5.4 User Story #4

The Garage Door view will be loaded when a user clicks an online Garage Door object from the Main View. This view, like all Controller Views, will have a navigation bar with elements to go back to the main view or go to the status view. The view will have a percentage bar that will update the user on that status of the door. It will also have a dynamic button to open or close the door. The text of the button will be based on the state of the door. If the device fails to load, or if the device has an error, an alert will pop up on the screen directing the user to go to the status view. If an error occurs, the UI elements will lock until the error is resolved.

6.0 Research or Proof of Concept Results

Generic, “off-the-shelf” garage door openers and sprinkler systems both have a generalized, external-facing electronic interface which can be readily tapped into by our system. Although neither device is wireless, they are controlled entirely by external switching mechanisms and should be able to provide great examples of how a simple system can be created on our platform. These can be easily controlled with a cheap microcontroller available at most electronic stores today, and with some inexpensive wireless chips we should be able to get these devices controlled completely wirelessly. If all other communication systems fail, we can use an Ethernet shield and some CAT5 cable to enable communication between our base station and the microcontroller.

We will be using school-issued laptops to emulate the base station, rather than purchasing a separate unit. If there is an excess of time at the end of the project, a separate unit may make a nice deliverable in addition to the base station software, but for the present it makes more sense to prototype on a local tablet.