CPRE 491 WEEKLY REPORT 03

Project Molecule

20 – 26 September 2016

May1739
may1739@iastate.edu

Dr. Arun Somani

Ryan Wade – Team Leader

Nathan Volkert – Communications Lead

Daniel Griffen – Key Concept Holder

Alex Berns – Webmaster & Scribe

1 CONTENTS

2 Weekly Summary 2
3 Past week accomplishments 2
4 Individual contributions 2
5 Comments and extended discussion 3
5.1 Use Cases: 3
5.2 Data Types 5
5.3 API 6
6 Plan for coming week: 7
7 Summary of weekly advisor meeting 7

2 WEEKLY SUMMARY

This week we met with our advisor and worked through several use cases for our project. We started brainstorming the architecture of our API

3 PAST WEEK ACCOMPLISHMENTS

All Members:

- Career Fair – This interfered with our regular weekly meeting
- Advisor Meeting

Ryan Wade:

- Worked on preliminary API
- Worked on Use Case for Light/Outlet Control

Nathan Volkert:

- Worked on Use Case for IFTTT (ifthisthenthat)
- Worked on Documentation/Forms

Daniel Griffen

- Worked on Use Case for media server

Alex Berns

- Worked on Use Case for Scheduler

4 INDIVIDUAL CONTRIBUTIONS

NAME	Hours this week	Hours cumulative
Ryan Wade	3	10
Nathan Volkert	2	7
DanielGriffen	2	7
Alex Berns	2	7

5 COMMENTS AND EXTENDED DISCUSSION

5.1 USE CASES:

- Media Server – play

  Requires file system access

  - File system access must be restricted to a specific set of folders

  Allow other extensions to request media file streams

  - Network communication between extensions should be controlled through the main binary
- Light/Outlet Control – on

  Requires GPIO send/receive control
  - Should abstract general communications drivers (I2C, SPI, etc) as extensions
  - Wrap all system resource calls so that we can instill permissions

  Allow other extensions to control the lights

  - Provide a way to advertise what public methods are available or to request what extensions can be controlled with said messages

  UI

  - Could provide a UI component for interacting with the lights
- Scheduler

  Requires Local Storage for events
  - Need to store events on multiple nodes, the event will come from a web interface node

  Data Type
  - Event(Title: String, Date: Date, Desc: String, Location: String, Notification Time: Date, Notification Action: Method, Participants: String)

  Actions

  Add New Event

  Input: New Event Object

  Output: True if successfully added

  Delete Event

  Input: Event Object

  Output: True if successfully changed

  Modify Event

  Input: New Event Object

  Output: True if successfully changed

  Notification Time Arrived

  Input: N/A

  Output: Notification Flag

  Event Arrived

Input: N/A
Output: Notification Flag

List Events
Input: New Event Object
Output: List of Events

List Free Times
Input: N/A
Output: List of free Times

- ifthisthanthat extension
  - IFTTT triggers when changes are detected and can notify other nodes
    - Should be able to communicate with other nodes on event
    - Should not override other communication going on

  IFTTT should make automation of the system easier

  Web-hooks

  Possibly look into web-hooks to use IFTTT in the UI also

5.2 DATA TYPES

a. sender/recipient ExtensionID
    i. Combines an application specific and an installation specific token to form a globally unique key (Node ID + Extension ID)
b. message
    i. Message Type
    ii. Data
c. messagePromise/resourcePromise
    i. onResolve()
    ii. onTimeout()
    iii. Status
    iv. (Data/SystemResource/Extension ID)
d. resourceType
    i. GPIO pins or Network Socket
e. systemResource
    i. wraps system resources so that we can apply permissions to
f. messageDescriptors
    i. describes what messages are publicly/privately available and their types.

5.3 API

g. private System API
    i. Register(ApplicationID, messageDescriptors)
        returns messagePromise
        called when extension is first installed

h. Public System object API
    i. **static** getPermissions()
        returns resourcePromise
        Used to get System Object Instance from internal Binary
        Ties process id to Extension ID
    ii. getResource(resourceType)
        returns resourcePromise
    iii. releaseResource(systemResource)
        returns resourcePromice
    iv. Message(recipient, message)
        returns messagePromise
        used to message other nodes/extensions
    v. findExtensionsByMessageType(messageType)
        returns messagePromise

i. Extension API
    i. onWake()
        Event: prepare for activity
    ii. onSleep()
        Event: prepare for sleep
    iii. orTerminate()
        Event: prepare to stop
    iv. onMessage(ExtensionID, recipient, message)
        Used by system to message the extension

6 PLAN FOR COMING WEEK:

All Members:

- Continue refining API support (layered approach?)
- Define Lifecycle of Extensions
- Define Lifecycle of Nodes

7 SUMMARY OF WEEKLY ADVISOR MEETING

Met for the first time, discussed the overview of the project and set up weekly advisor meetings each Wednesday.